AD-757 090

RESEARCH IN STORE AND FORWARD COMPUTER NETWORKS

Howard Frank

Network Analysis Corporation

Prepared for:

Advanced Research Projects Agency

December 1972

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R&D.

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)
Network Analysis Corporation
Beechwood, Old Tappan Road
Glen Cove, New York 11542

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP
None

3. REPORT TITLE
Final Report for the Project "Research in Store-and-Forward Computer Networks"

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Final Technical Report, 13 October 1969--12 October 1972

5. AUTHOR(S) (Last name, first name, initial)
Network Analysis Corporation

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December, 1972	120 /25	12

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	ARPA FINAL REPORT No. 1
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. AVAILABILITY/LIMITATION NOTICES
This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
None	Advanced Research Projects Agency, Department of Defense

13. ABSTRACT
This document describes research efforts to determine: 1)the most economical configurations to meet growth requirements in the ARPANET, 2) properties of packet switched computer communication networks, 3) analysis and design techniques for large scale networks, 4) cost/throughput/reliability characteristics of large packet switched networks for potential application to Defense Department computer communication requirements. The heart of the research program has been the dual attack on basic network theoretical problems and the development of computational techniques for handling large network structures. Results on properties of the ARPANET, properties of large networks, and new computational techniques are described for subjects as traffic sensitivity, peak bandwidth, reliability, cost and throughput and routing.

14. KEY WORDS
Computer networks, throughput, cost, reliability, survivability, ARPA Computer Network, store-and-forward, packet switching.

UNCLASSIFIED
Security Classification

FINAL TECHNICAL REPORT

(13 October 1969 - 12 October 1972)

FOR THE PROJECT

"RESEARCH IN

STORE AND FORWARD COMPUTER NETWORKS"

Principal Investigator
and Project Manager:

HOWARD FRANK (516) 671-9580

ARPA Order No. 1523

Contractor: Network Analysis Corporation

Contract No. DAHC 15-70-C-0120

Effective Date: 13 October 1969

Expiration Date: 12 October 1972

Sponsored by

Advanced Research Projects Agency

Department of Defense

SUMMARY

Technical Problem

Network Analysis Corporation's contract with the Advanced Research Projects Agency has had the following objectives:

- To determine the most economical configurations to meet growth requirements in the ARPANET.
- To study the properties of packet switched computer communication networks.
- To develop techniques for the analysis and design of large scale networks.
- To determine the cost/throughput/reliability characteristics of large packet switched networks for potential application to Defense Department computer communication requirements.

General Methodology

The heart of the research program has been the dual attack on basic network theoretical problems and the development of computational techniques for handling large network structures. For example, efficient reliability analysis procedures were first developed and these procedures were then used to study both ARPANET reliability and large network reliability.

Technical Results

The technical results can be grouped into three areas: properties of the ARPANET, properties of large networks, and computational techniques.

Properties of the ARPANET

Traffic sensitivity: It was shown that the performance of the ARPANET is highly insensitive to input traffic distributions, and that the network retains its high throughput capabilities even though these distributions fluctuate widely.

Peak bandwidths: It was shown that the high peak bandwidths achievable in the network are obtained at virtually zero incremental cost.

Reliability: The network was shown to have high reliability with both nodes and links at present equipment reliabilities. A detailed throughput reliability analysis of the ARPANET considering element failures, traffic requirements, routing and acceptable delays shows that the network is highly adaptable to component failures.

Cost: The network has exhibited substantial economies of scale as new nodes have been added and as the capacity of a particular network with a given number of nodes is increased. At a recent stage in the network's growth, the network was shown to be within 1% of a theoretical (but physically unachievable) optimum cost.

Properties of Large Networks

Cost and throughput: Cost and throughput characteristics were derived for a family of networks containing 20, 40, 60, 80, 100, and 200 nodes distributed across the United States. It was shown that networks of this size exhibit economies of scale, deliver required performance using current ARPANET technology, and provide costs comparable to the current network.

Reliability: A study of the tradeoffs between network size, network connectivity and component reliability was performed. This study indicated that reliability will be a major design constraint for large network design but that for the sizes of networks considered it is a completely controllable parameter using the distributed ARPANET technology.

Computational Techniques

Routing: Two basic classes of routing procedures were developed. The first is capable of finding optimal routes for traffic and hence is capable of deriving upper bounds on network performances. The second class of procedures were developed for use within the design process. This class incorporates heuristic and analytical techniques to produce flows close to optimal with very small amounts of computation.

Reliability: Major computational improvements for large network reliability analysis were developed. One such technique was established to be more than 1000 times more efficient than conventional simulation procedures. In addition, a new method for

reliability analysis which uses a recursive technique has been developed to handle a large class of networks composed of loops and trees. This method allows a wide variety of reliability criteria to be evaluated simultaneously at a small fraction of the cost of previously known methods.

Basic Network Algorithms: New and improved techniques for a number of fundamental network computations were developed. These include the calculation of network components, the generation of shortest paths, and the generation of minimum spanning trees. These computations are basic ingredients in many large scale network algorithms and consequently the size of the tractable problems using these techniques has been extended.

Department of Defense Implications

The Defense Department has vital need of highly reliable and economical communications. The contract studies establish the substantial cost advantages of packet switched "ARPANET like" computer communication systems with as many as 200 nodes. The reliability studies developed the tools needed to first analyze and then control the reliability of these structures. The computational improvements increase the size of networks that can be analytically studied and hence optimized.

Implications for Further Research

Communication networks for meeting Defense Department requirements involve huge network structures that present techniques are not yet adequate to handle. The research during the contract period identified some of the unique and highly desirable properties of distributed computer communication systems and demonstrated the performance trends of these networks as they increase in size. The studies show that for very large networks, cost-reliability considerations must be given equal importance to cost-throughput considerations. This means that there will be a need to develop dramatically different network design procedures to insure availability of resources in a large network. The requirements of the new procedures, while not yet well defined, indicate that computation breakthroughs for a number of basic network problems will be necessary. The potential of these networks to the DOD establishes a high priority in seeking these breakthroughs.

TABLE OF CONTENTS

1. INTRODUCTION AND SUMMARY *

Network Analysis Corporation's contract with the Advanced Research Projects Agency has had the following objectives:

- To determine the most economical configurations to meet growth requirements in the ARPANET.
- To study the properties of packet switched computer communication networks.
- To develop techniques for the analysis and design of large scale networks.
- To determine the cost/throughput/reliability characteristics of large packet switched networks for potential application to Defense Department computer communication requirements.

The heart of the research program has been the dual attack on basic network theoretical problems and the development of computational techniques for handling large network structures. For example, efficient reliability analysis procedures were first developed and these procedures were then used to study both ARPANET reliability and large network reliability.

---

* Throughout this report, a reference to Report "I" is a reference to NAC's "I-th" Semiannual Report to ARPA.

The technical results discussed in this report are grouped into three sections: properties of the ARPANET, properties of large networks, and computational technniques. The report summarizes the significant developments of the research effort. These include:

Properties of the ARPANET

Traffic sensitivity: It was shown that the performance of the ARPANET is highly insensitive to input traffic distributions, and that the network retains its high throughput capabilities even though these distributions fluctuate widely.

Peak bandwidths: It was shown that the high peak bandwidths achievable in the network are obtained at virtually zero incremental cost.

Reliability: The network was shown to have high reliability with both nodes and links at present equipment reliabilities. A detailed throughput reliability analysis of the ARPANET considering element failures, traffic requirements, routing, and acceptable delays shows that the network is highly adaptable to component failures.

Cost: The network has exhibited substantial economies of scale as new nodes have been added and as the capacity of a particular network with a given number of nodes is increased. At a recent stage in the network's growth, the network was shown to be within 1% of a theoretical (but physically unachievable) optimum cost.

Properties of Large Networks

Cost and throughput: Cost and throughput characteristics were derived for a family of networks containing 20, 40, 60, 80, 100, and 200 nodes distributed across the United States. It was shown that networks of this size exhibit economies of scale, deliver required performance using current ARPANET technology, and provide costs comparable to the current network.

Reliability: A study of the tradeoffs between network size, network connectivity and component reliability was performed. This study indicates that reliability will be a major design constraint for large network design but that for the sizes of networks considered it is a completely controllable parameter using the distributed ARPANET technology.

Computational Techniques

Routing: Two basic classes of routing procedures were developed. The first is capable of finding optimal routes for traffic and hence is capable of deriving upper bounds on network performance. The second class of procedures was developed for use within the design process. This class incorporates heuristic and analytical techniques to produce flows close to optimal with very small amounts of computation.

Reliability: Major computational improvements for large network reliability analysis were developed. One such technique was established to be more than 1000 times more efficient than conventional simulation procedures. In addition, a new method for reliability analysis which uses a recursive technique has been developed to handle a large class of networks composed of loops and trees. This method allows a wide variety of reliability criteria to be evaluated simultaneously at a small fraction of the cost of previously known methods.

Basic Network Algorithms: New and improved techniques for a number of fundamental network computations were developed. These include the calculation of network components, the generation of shortest paths, and the generation of minimum

Computational Techniques

Routing: Two basic classes of routing procedures were developed. The first is capable of finding optimal routes for traffic and hence is capable of deriving upper bounds on network performance. The second class of procedures was developed for use within the design process. This class incorporates heuristic and analytical techniques to produce flows close to optimal with very small amounts of computation.

Reliability: Major computational improvements for large network reliability analysis were developed. One such technique was established to be more than 1000 times more efficient than conventional simulation procedures. In addition, a new method for reliability analysis which uses a recursive technique has been developed to handle a large class of networks composed of loops and trees. This method allows a wide variety of reliability criteria to be evaluated simultaneously at a small fraction of the cost of previously known methods.

Basic Network Algorithms: New and improved techniques for a number of fundamental network computations were developed. These include the calculation of network components, the generation of shortest paths, and the generation of minimum

spanning trees. These computations are basic ingredients in many large scale network algorithms and consequently, the size of the tractable problems using these techniques has been extended.

The Defense Department has vital need of highly reliable and economical communication. The contract studies established the substantial cost advantages of packet switched "ARPANET like" computer communication systems with as many as 200 nodes. The reliability studies developed the tools needed to first analyze and then control the reliability of these structures. The new computation improvements increase the size of networks that can be analytically studied and hence optimized.

Communication networks for meeting Defense Department requirements involve huge network structures that present techniques are not yet adequate to handle. The research identified some of the unique and highly desirable properties of distributed computer communication systems and demonstrated the performance trends of these networks as they increase in size. The studies show that for very large networks, cost/reliability considerations must be given equal importance to cost/throughput considerations. This means that there will be a need to develop dramatically different network design

procedures to insure availability of resources in a large network. The requirements of the new procedures, while not yet well defined, indicate that new computation breakthroughs for a number of basic network problems will be necessary. The potential of these networks to the DOD establishes a high priority in seeking these breakthroughs.

2. PROPERTIES OF THE ARPANET

2.1 ARPANET Topological Design

During the contract period, many network optimizations were performed to introduce new nodes into the ARPANET, to test and improve overall network economy and reliability, and to study the economic and growth characteristics of the ARPA network.

Since initially there was no clear knowledge of the total traffic the network would have to accommodate, the network was first constructed with enough capacity to accommodate any reasonable traffic requirements. At the initial stages of the design, the "two-connected" reliability constraint forced the network throughput to be in the range 10-15 KBPS/node since two communication paths between every pair of IMPs is needed. As new IMPs were added to the network, the capacity is being systematically reduced until the traffic occupies a substantial fraction of the network's total capacity. At present, it appears that this point will be reached in early 1974. The network's capacity will then be increased to maintain a desired percentage of loading. To insure that this process can be efficiently performed, each basic configuration is designed so that additional links can be added to economically increase network throughput.

If the locations of all network nodes are known in advance, it is clearly most efficient to design the topological structure as a single global effort. However, in the ARPANET, as in most actual networks, node locations are added and modified on numerous occasions. On each such occasion, the topology could be completely reoptimized to determine a new set of link locations.

In practice, however, there is a long lead time between the ordering and the delivery of a link and major topological modifications cannot be made without substantial difficulty. It is therefore prudent to add or delete nodes with as little disturbance as possible to the basic network structure consistent with overall economic operation.

At approximately 26 nodes, the growth pattern within the net made it desirable to implement some fundamental changes in network structure. The original net expanded eastward from a 4-node configuration on the West Coast. Because of this origination, the West Coast had somewhat more capacity than other parts of the country. Also, because of the excellent relative location of the UTAH node, two of the three planned cross country paths utilized this node thus creating a great dependence in the enlarged net. Finally,

the expanded net has a number of new nodes in the Washington, D.C. area. A redesign of the network taking advantage of these facts was able to reduce cost while simultaneously increasing network throughput and reliability. To test the overall economy of the new design, an additional design was generated under the assumption that all 26 nodes were to be interconnected into a network at the same time, with no restrictions on link locations. This design, which represents a "global" optimization, believed to be optimal under uniform traffic requirements, was only 1% lower in cost per year. Thus, the actual 26-node ARPANET design is within $10,000 (less than $400/node/year) of a theoretically globally optimum but unrealizable solution.

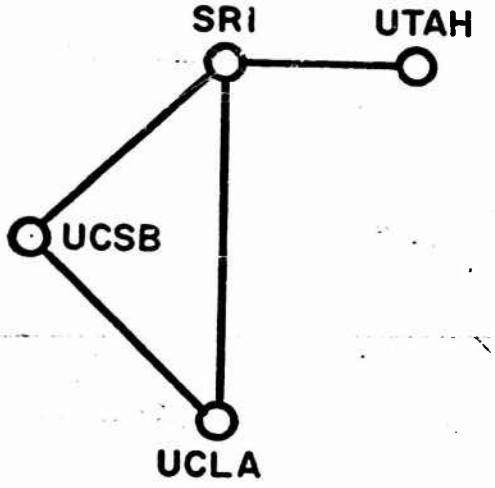
Current network designs contain on the order of 40 nodes. The evolution to this stage is summarized in Table 2.1. This table shows the performance and economic trends indicated by the growth of the net from ten nodes to the presently planned net. During the growth of the net, the link cost per node has been reduced from $44,000/node/year for the 15-node net to $25,550/node/year for the 40-node network while the design throughput level has been reduced only from 10.7 KBPS/node to 7.3 KBPS/node for the same networks. The line cost per

thousand packets of transmitted data has been relatively constant at 7 cents (assuming 24 hour per day, 7 days per week operation). Some representative ARPANET designs are shown in Figures 2.1a-2.1f.

TABLE 2.1

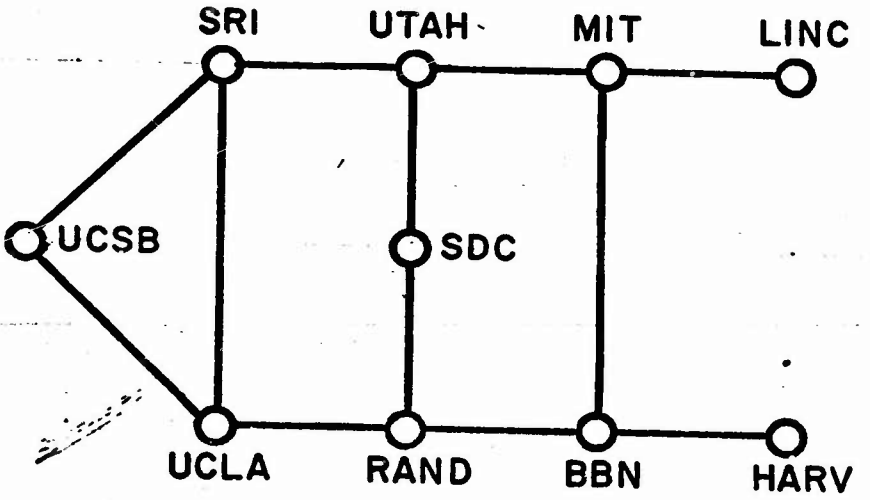
NETWORK LINE COSTS

Number of Nodes	Yearly Line Cost (K$)	Throughput (Uniform Traffic) KBPS/Node	Throughput (Uniform Traffic) KPacket/Day/Node	Line/Cost Node (K$)	Line/Cost KPacket (cents)
14	605	10.5	1449	43.2	8
15	659	10.7	1478	43.9	8
18	792	12.2	1684	44.0	7
21	825	10.6	1463	39.3	7
23	849	10.2	1408	36.9	7
24	860	9.5	1311	35.8	7
26	810	9.9	1366	31.2	6
30	859	8.7	1201	28.6	7
33	886	8.0	1104	26.8	7
39	1,016	7.5	1035	26.1	7
40	1,022	7.3	1007	25.6	7



4 NODE NETWORK 12-1-69

FIGURE 2.1a



10 NODE NETWORK 7-1-70

FIGURE 2.1b

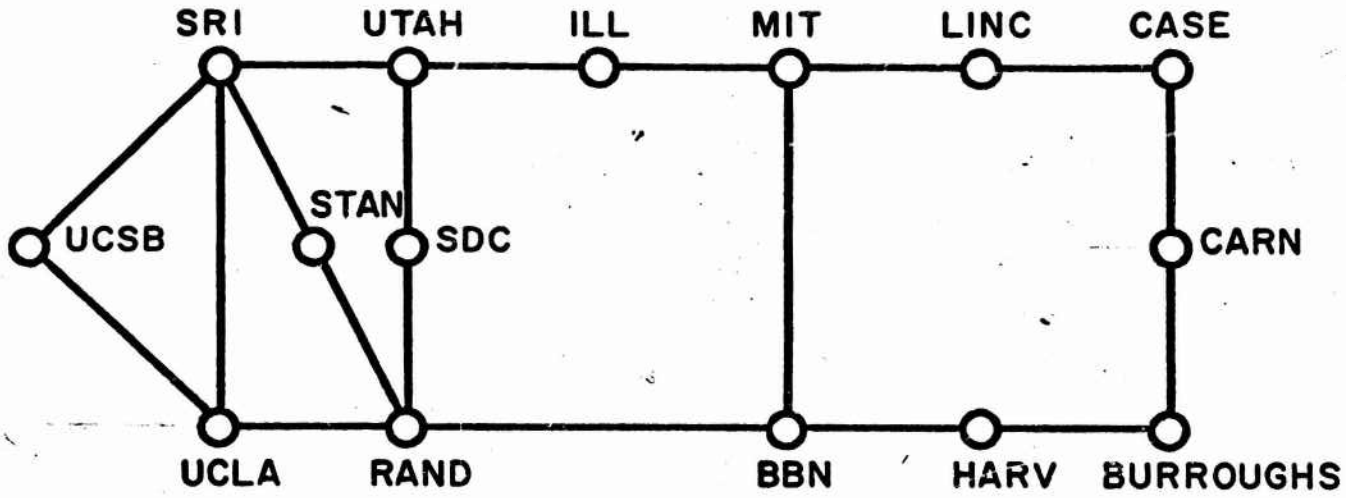


FIGURE 2.1c

15 NODE NETWORK 3-1-71

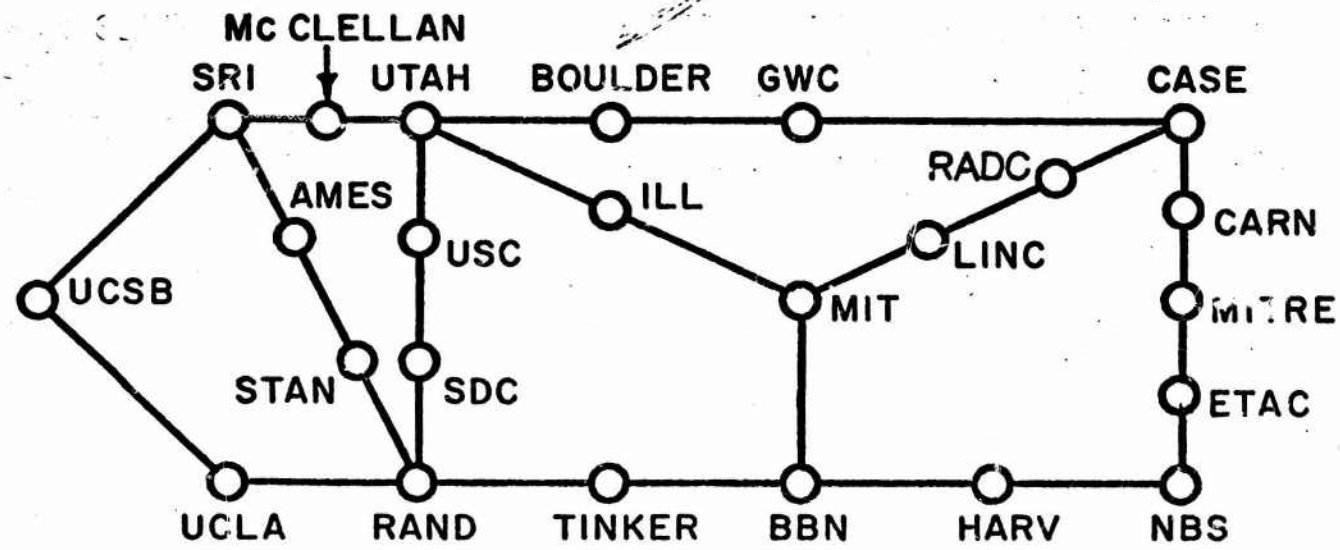


FIGURE 2.1d

24 NODE NETWORK 4-1-72

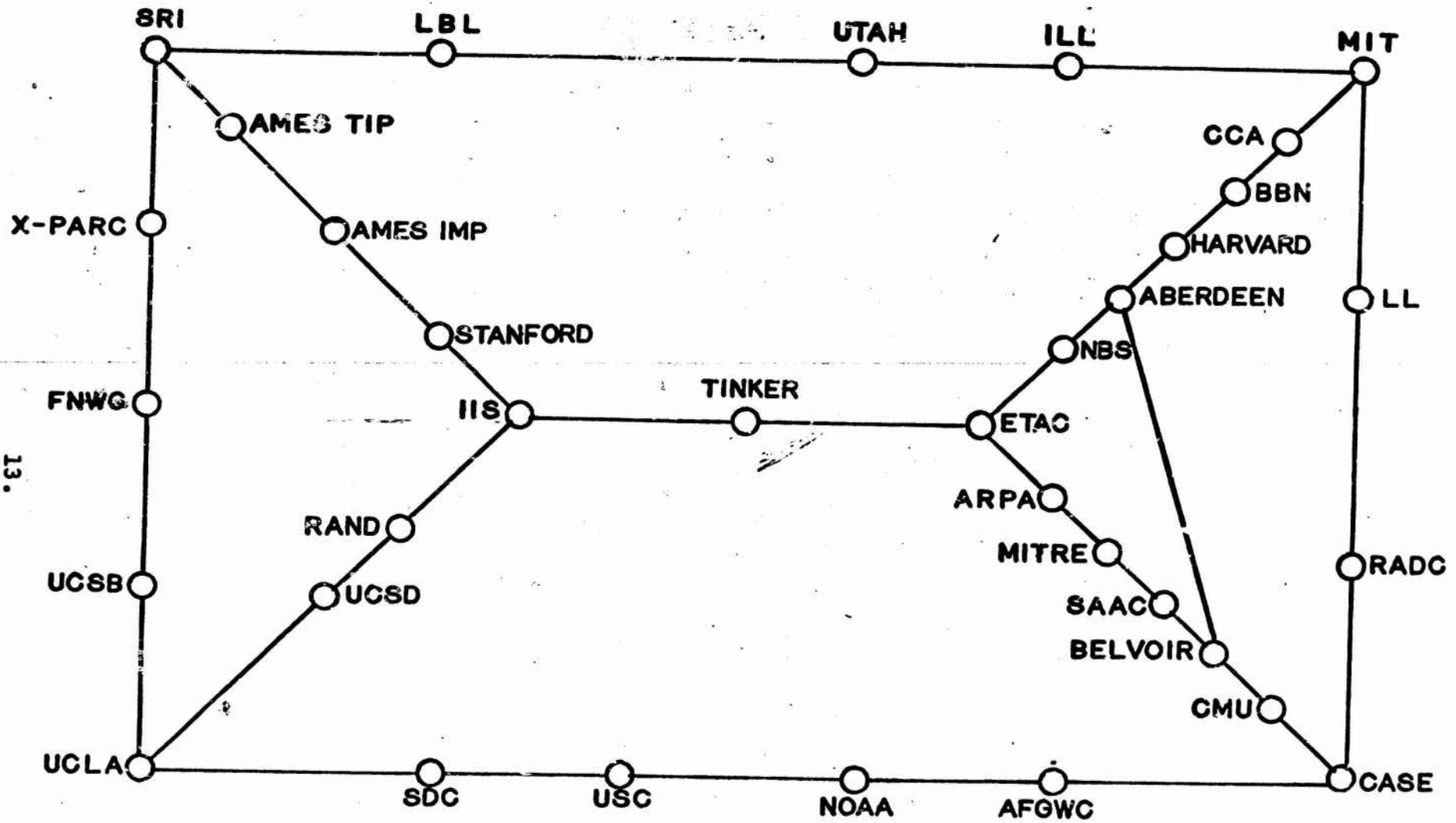


FIGURE 2.1e

33 Node Network

LATE 1972 NETWORK

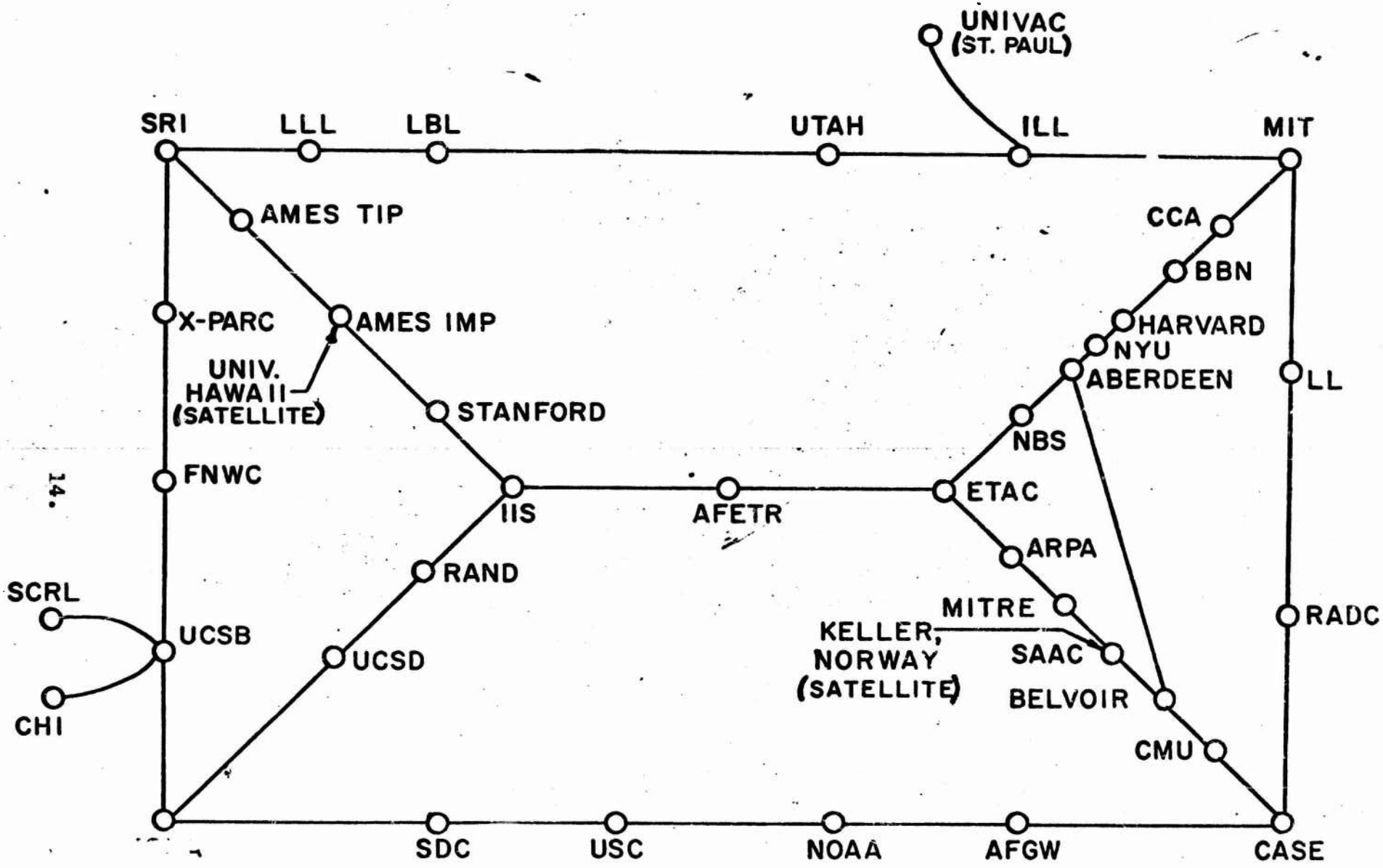


FIGURE 2.1f

PROJECTED EARLY 1973 NETWORK

2.2 Traffic Sensitivity

Any network design depends on the forecast of the traffic distribution. If this forecast is inaccurate, it can be expected that inefficiencies in performance will occur. In our case, these inefficiencies could be excessive time delay at the desired throughput levels, or equivalently, lower throughputs at saturation. In order to determine the effect of traffic variations on time delay and throughput, several simulation and analysis experiments were conducted.

The basic ARPA configurations were derived with essentially equal traffic between all node pairs. All node pair traffic levels are increased at the same rate until time delay equals 0.2 seconds.

The average traffic per node at 0.2 second average delay is defined to be the network's throughput. The following experiment can be performed to examine the effect of widely varying node output rates.

1. A random number Tr(I) bounded by the constants TRL and TRU (i.e., $TRL \leq Tr(I) \leq TRU$) is selected uniformly at random for node I for I=1, ..., N. This random number is set equal to the total output of node I.

2. For each node I, N-1 non negative random numbers $K(I,I_1)$, $K(I,I_2)$, ..., $K(I,I_{N-1})$ are generated uniformly at random . The traffic from node I to node J is set equal to

$$TR(I) = K(I,I_J) / \sum_{j=1}^{N-1} K(I,I_J)$$

3. The routing algorithm is applied and the average number of bits/second determined.

4. Steps 1-3 are repeated until a statistically valid sample is obtained.

Steps 1-3 were performed with a typical 12-node network and an 18-node network. In the experiments, TRL ranged from 138 to 271 KPackets/Day and TRU ranged from 2622 to 5244 KPackets/Day and 200 sets of flows were generated for the 12-node network and 100 sets for the 18-node network.

Tables 2.2 & 2.3 summarize the data collected. The effect of traffic variations for the two networks can be readily investigated.

In both the twelve and eighteen node cases, throughput for uniform traffic distributions were from 10% to 13% higher than the average throughputs for the random samples. In addition, for both networks, more than 75% of the random

TABLE 2.2
12 NODE NETWORK

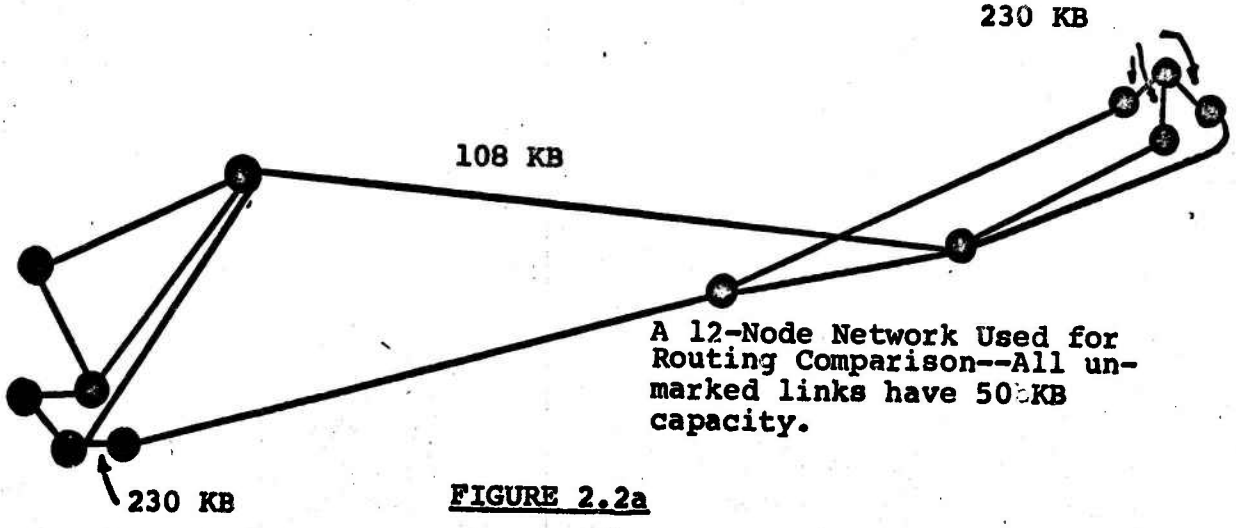
Average Throughput (KPackets/Day/Node)	Number of Traffic Patterns
1932	1
2070	3
2208	5
2346	16
2484	24
2622	44
2760	39
2898	44
3036	13
3174	8
3312	3

Sample mean = 2760 KPackets/Day/Node
Sample Deviation = 276 KPackets/Day/Node

TABLE 2.3
18 NODE NETWORK

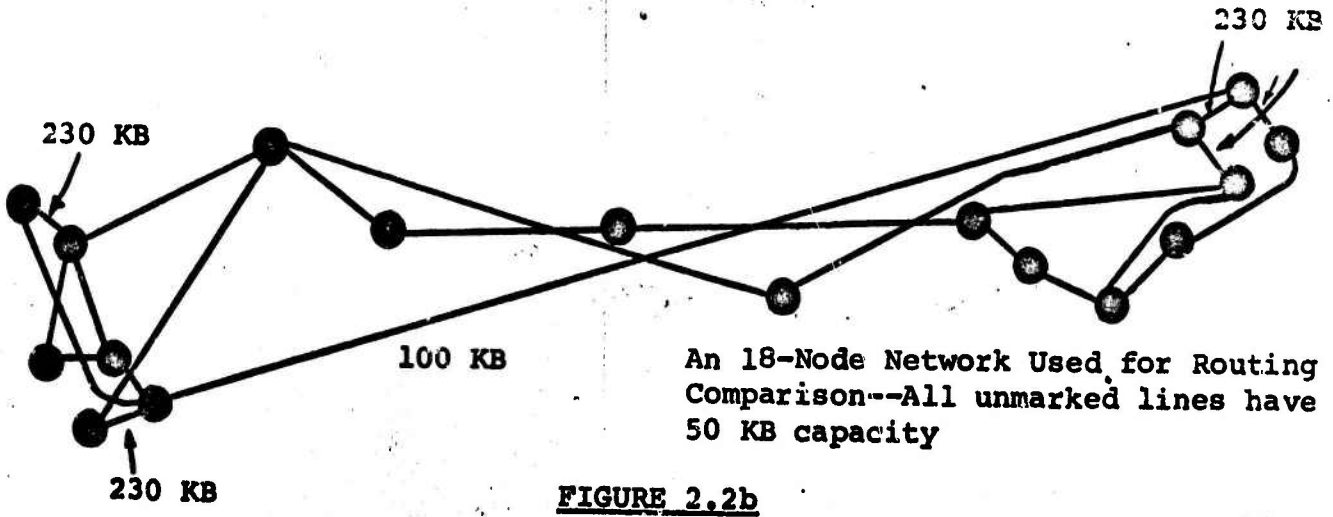
Average Throughput (KPackets/Day/Node)	Number of Traffic Patterns
1794	1
1932	5
2070	16
2208	19
2346	33
2484	18
2622	5
2760	3

Sample mean = 2346 KPackets/Day/Node
Sample Deviation = 205 KPackets/Day/Node



A 12-Node Network Used for Routing Comparison--All unmarked links have 50 KB capacity.

FIGURE 2.2a

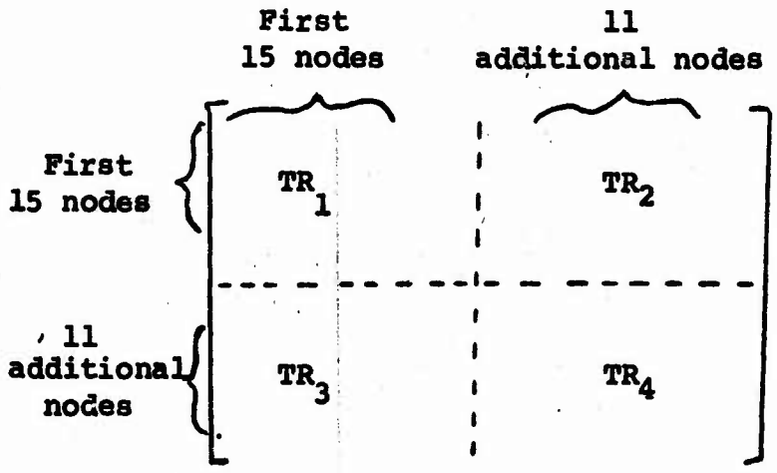


An 18-Node Network Used for Routing Comparison--All unmarked lines have 50 KB capacity

FIGURE 2.2b

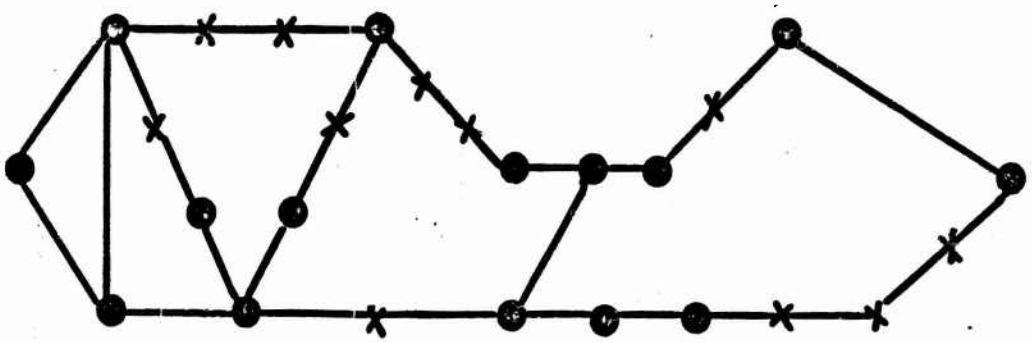
cases have average throughputs within 17% of the uniform traffic throughput. These numbers demonstrate that the degradation in performance caused by large variations in flow requirements is not great. In other words, the routing procedure is able to efficiently handle significantly different input flows without excessive time delay.

Another set of sensitivity analyses were performed on two 26 node ARPANET configurations. This set postulated highly unequal traffic distributions where the response at selected locations generated up to 10 return packets for each packet received. The locations of the points generating such high traffic was selected from the basic ARPANET resources at UCLA, UCSB, SRI, BBN, MIT, AMES, and NOAA. The precise system requirements were then widely varied from the uniform design distribution. The results are summarized in Figures 2.3a - 2.3h . Traffic for these systems are generated by partitioning the nodes into 4 groups identified by ●, ■ , x and ▲. The nodes identified by ● and ■ constitute the first 15 nodes added to the network. Flow requirements among these nodes is adjusted independent of the other nodes. The traffic matrix $TR = [tr_{i,j}]$ where $tr_{i,j}$ is the flow from node i to node j is partitioned in the following manner.



The traffic in each of the four submatrices is adjusted independently. Using varying traffic patterns the flows in TR_1 are selected to yield a specified set of flows between the 15 nodes. The maximum amount of traffic that can then be sent from the 11 nodes is then calculated under several different assumptions about traffic patterns.

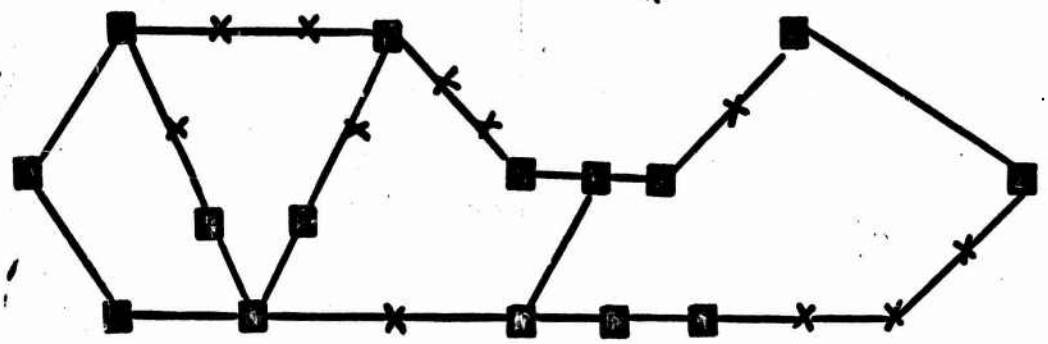
The results of the above analysis again indicate that large traffic requirement variations do not result in substantial changes in network performance. Thus, for the two 26-node network configurations illustrated, the first network has an average capacity of 925 KPackets/Day/Node for the patterns examined; the lowest throughput pattern measured is 15% beneath the average and the highest is 10% above the average. The second configuration has an average throughput of 1082 KPackets/Day/Node; the lowest throughput pattern measured is 9% beneath this average the the highest is 20% above the average.



Traffic Pattern: Equal traffic between ● nodes, equal traffic from all x nodes.

KPackets/Day/Node Between ● Nodes	Maximum KPackets/Day/Node From x Nodes	Total KPackets Day/Node
1694	0	977
1542	250	995
1401	478	1011
1239	712	1016
906	906	906

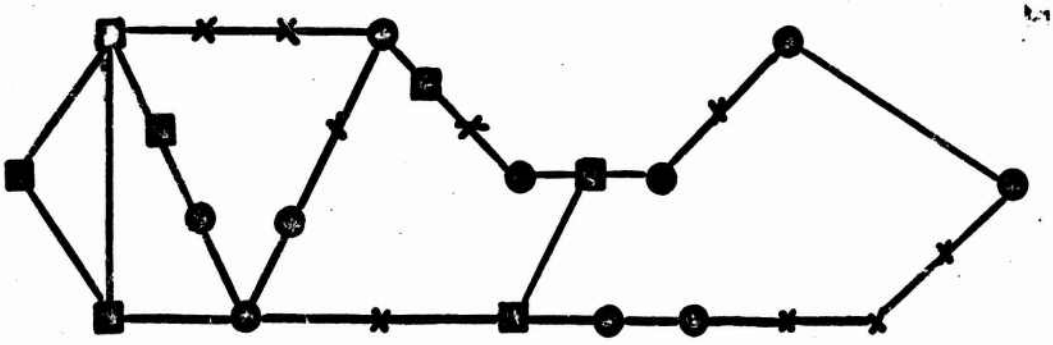
FIGURE 2.3a



Traffic Pattern: 10 units of traffic from ■ nodes to x nodes, in response to each unit into ■ ; from each x node to all nodes, 1 unit of traffic between ■ nodes.

KPackets/Day/Node Between ■ Nodes	Maximum KPackets/Day/Node from x Nodes	Total KPackets Day/Node
1576	193	991
1432	373	984
1266	557	966

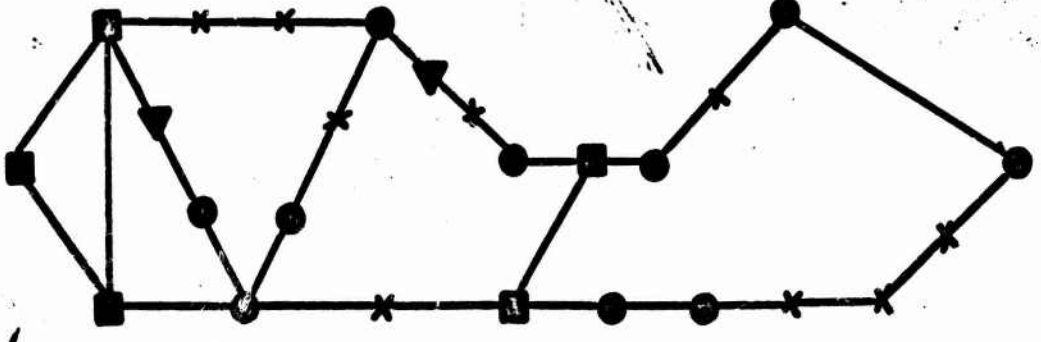
FIGURE 2.3b



Traffic Pattern: 10 units of traffic from ■ to x nodes for each unit in; equal traffic originated at every node and destined for every other node.

KPackets/Day/Node: 783

FIGURE 2.3c



Traffic Pattern: 10 units from each ■ node to each ● node for each unit in; 10 units from each ▲ node to each x node for each unit in; 1 unit originated from each node to each other node.

KPackets/Day/Node Between ● and ■ nodes	Maximum KPackets/Day/Node from x and ▲ nodes	Total KPackets Day/Node
1410	0	813
1371	70	821
1239	294	839

FIGURE 2.3d

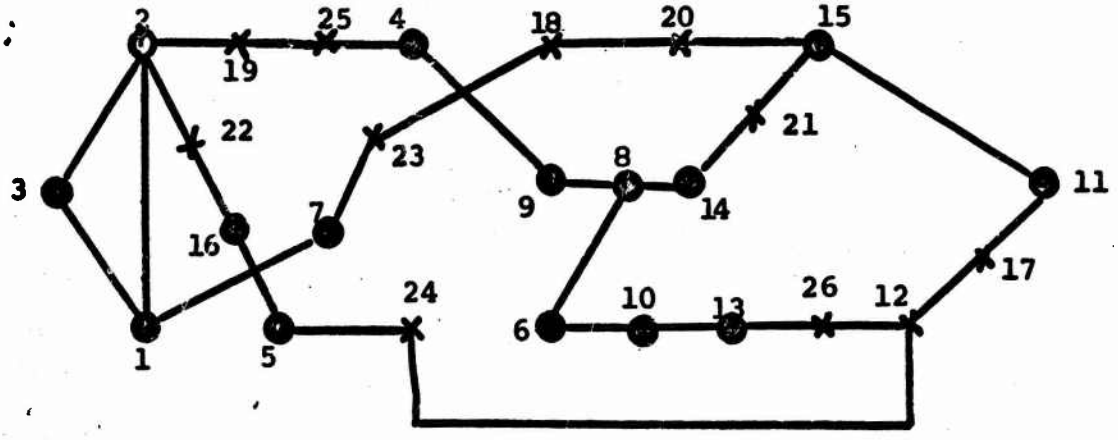
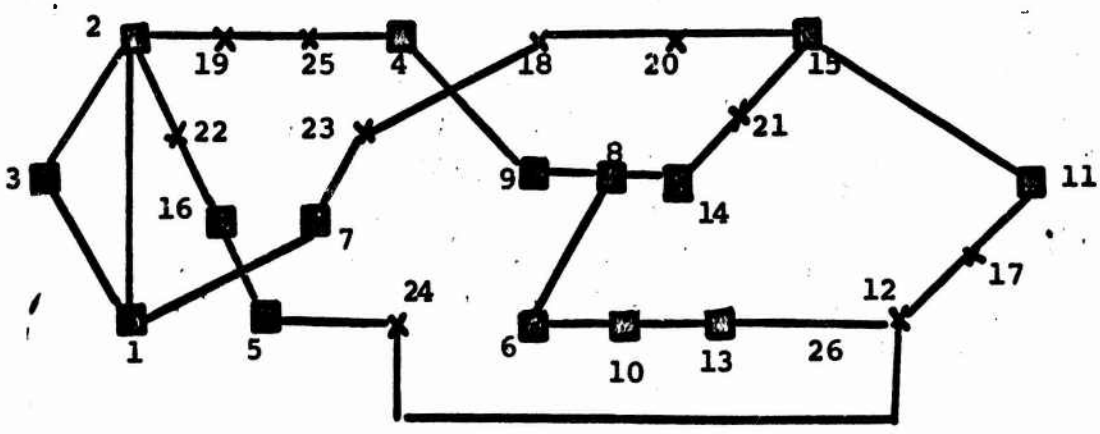


FIGURE 2.3e

Traffic Pattern: Equal traffic between ● nodes, equal traffic from all x nodes.

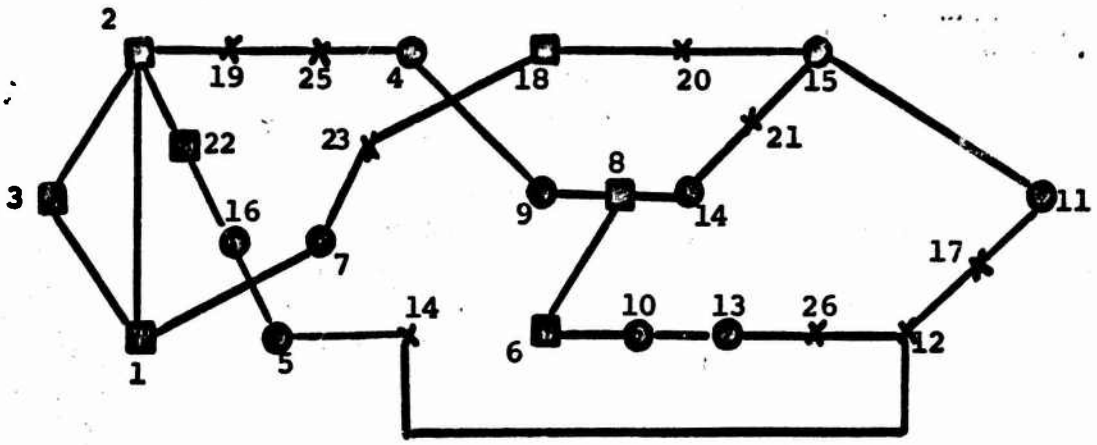
KPackets/Day/Node Between ● Nodes	Maximum KPackets/Day/ Node from x Nodes	Total KPackets/ Day/Node
2092	0	1206
1790	92	1071
1628	794	1275
1447	1052	1280
1301	1301	1301
1266	1311	1285



Traffic Pattern: 10 units of traffic from ■ nodes in response to each unit into ■ ; 1 unit of traffic from each x node to all nodes, 1 unit between ■ nodes.

KPackets/Day/Node Between ■ Nodes	Maximum KPackets/Day/Node from x Nodes	Total KPackets Day/Node
1751	593	1261
1594	793	1255
1416	996	1238
1253	1199	1231

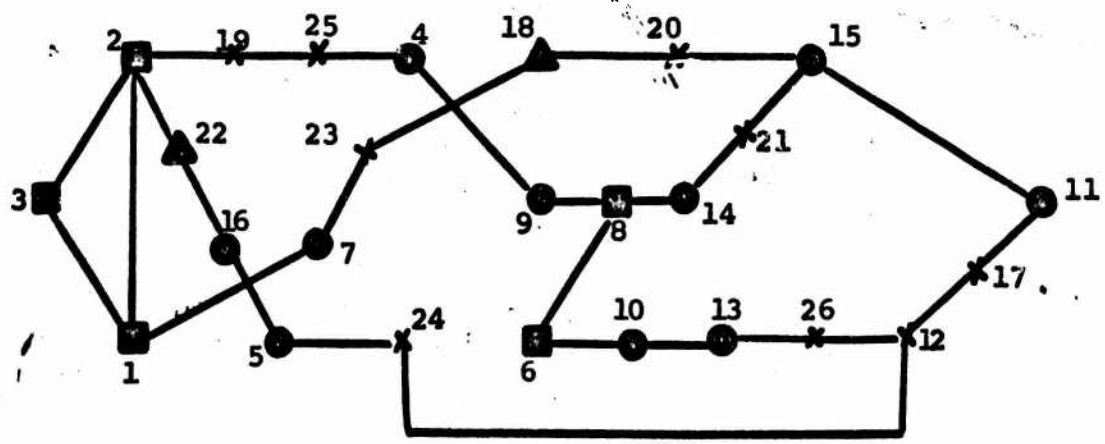
FIGURE 2.3f



Traffic Pattern: 10 units of traffic from ■ to x nodes for each unit in; equal traffic originated at every node and destined for every other node.

KPackets/Day/Node: 993

FIGURE 2.3g



Traffic Pattern: 10 units from each ● node to each ■ node for each unit in; 10 units from each ▲ node to each x node for each unit in; 1 unit originates from each node to each other node.

KPackets/Day/Node Between ● and ■ nodes	Maximum KPackets/Day/Node from x and ▲ nodes	Total KPackets Day/Node
1708	0	985
1594	222	1014
1446	492	1042
1266	756	1050

FIGURE 2.3h

2.3 Incremental Costs

Adding Capacity Without Adding Nodes

As part of the network design studies aimed at determining the cost-throughput characteristics of the ARPANET, numerous analyses of the cost of increasing the capacity of specific network designs from base traffic levels of 5-10 KBPS/Node to levels of 20 KBPS/Node or more have been performed. The relationships between line cost per year and throughput per node for three such studies are shown in Figure 2.4. The points on these curves were obtained by adding and deleting links from the basic network designs with the "subjective" constraint that as few modifications to the basic network structure are made. This goal is imposed because of the up to nine months' installation time for a major 50 KBPS line. This large lead time dictates that changes to the basic configuration be made infrequently.

It is evident from Figure 2.4 that by adding links to an existing ARPANET in as economical a fashion as possible, one obtains a linear relationship between increase in capacity and increase in cost. Furthermore, the dashed line indicates the boundary of the "economies of scale" region. That is,

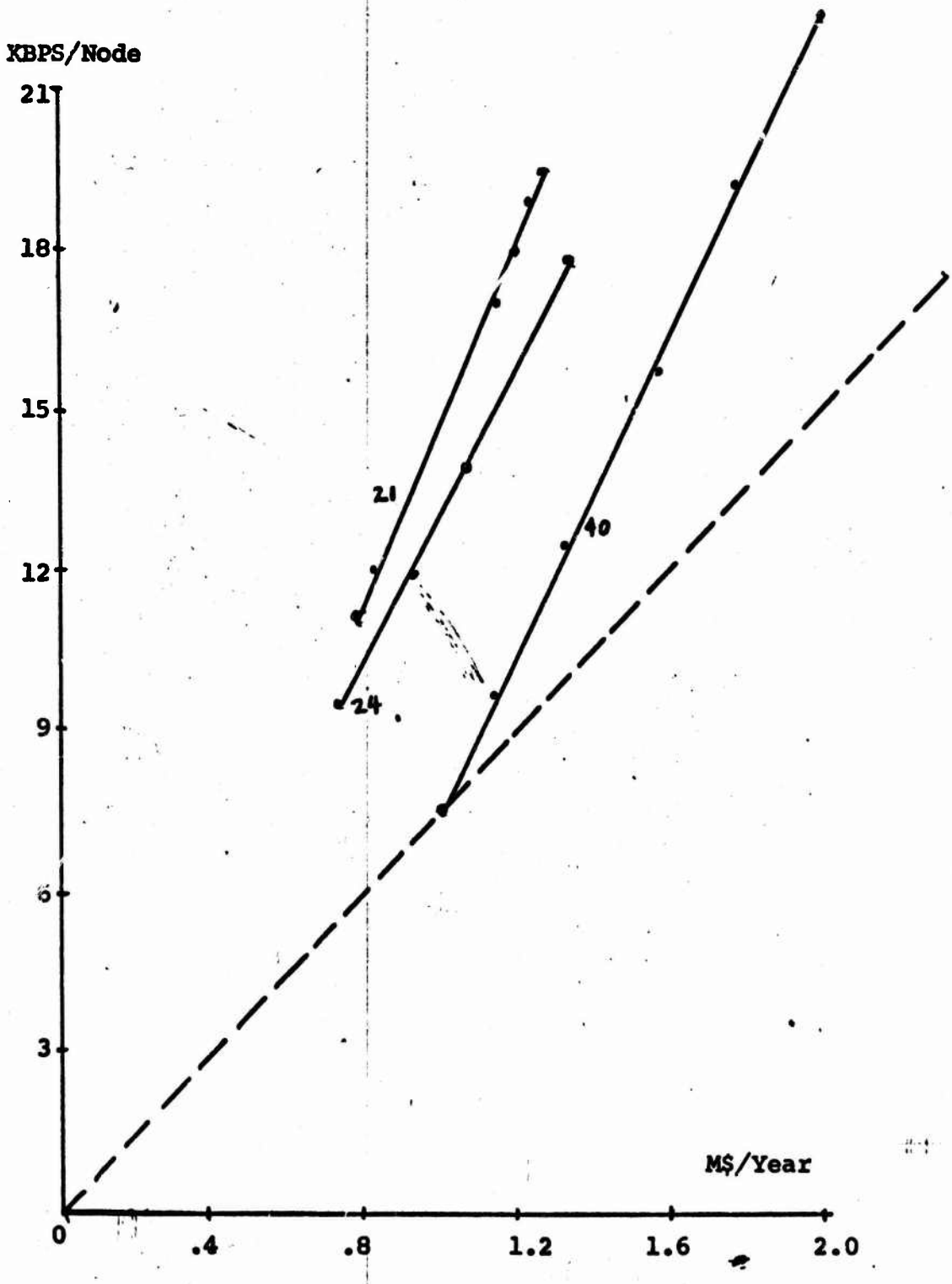


FIGURE 2.4

the linear relationship induced by the dashed line corresponds to a doubling of line cost for a doubling of capacity. Above the dashed line, where all three curves lie, the cost per bit decreases as the network capacity is increased. For example for the 40 node network, a threefold increase in capacity is achieved with only a twofold increase in cost of lines.

Adding Nodes

The rapid growth of the ARPANET creates the problem of equitably distributing the cost of the network over its community of users. There are two kinds of network users--ARPA contractors at universities and research centers and non-ARPA contractors who are joining the network to utilize resources such as the ILLIAC IV. The former group of users have been principally responsible for the growth and development of the network and the transition from an experimental project to a viable, economic tool broadly applicable to Defense Department communication problems. The latter user group is contributing the operating environment that will allow the network experiment to proceed from a specialized one to a general purpose one.

The problem of distributing costs between the two user groups was studied as follows (Report 4):

1) The 15 node network connecting the original set of ARPA contractors was first examined, and its throughput and cost determined subject to two types of traffic loads. The first assumes equal traffic between all pairs of nodes and the flow per node leading to an average time delay of 0.2 seconds is calculated. The second considers five nodes --BBN, UCLA, UCSB, SRI, and MIT--as network resources. Equal traffic from all nodes to these nodes is assumed but the return traffic from each of the five nodes to all nodes is assumed to be ten times as great as the forward traffic. The average throughput per node with this traffic pattern is then calculated at a 0.2 second time delay.

2) A 26 node network is derived by adding 11 new user nodes in the second category to the original 15 node net. The augmentation is made to minimize the incremental cost of adding the 11 nodes without regard to throughput. The throughput of the 26-node network is calculated under a variety of traffic conditions. In some of these calculations NASA and NOAA are considered to be additional network resources.

3) The planned 26 node network is analyzed in the same manner as the network discussed in 2).

The object of the three analyses is to compute the incremental cost to add the new nodes into the network, the cost required to provide service equivalent to that provided by the 15 node network to the original 15 nodes, the throughput that can be supplied to the new nodes if this should be required.

The results of the analyses indicate that:

1) A fixed line cost of approximately $16,500 per new node is directly attributable to the addition of the new nodes if the cost of the 15 node network is subtracted from the cost of the 26 node network.

2) Depending on the traffic pattern, the new nodes can transmit between 0 and 25 kilopackets per hour in the 26 node network if the original nodes receive throughput equal to that provided by the 15-node network. Additional throughput can only be provided to the new nodes by degrading the service to the original nodes or adding new communication links to the network.

Our previous analysis indicated that we may assume a linear cost to increase the capacity of the fixed network. In a 24 node network, a 1 million packet per day per node increase could be achieved for about $280,000 per year.

This yields a cost per thousand packets of about 3.2 cents. In a 40-node network it would cost about $1 million per year to achieve a 2 million packet per day/node increase in capacity. On a per kilopacket basis, this leads to about a 3.4 cents cost. These numbers are based on uniform traffic requirements and 24 hour/day, 7 days per week network operation. To compensate for non-uniform traffic effects, one might increase the cost per kilopacket to 4-5 cents and to compensate for the fact that the network will be fully loaded only a fraction of the time (say, 30-40 hours per week), one might multiply the per kilopacket cost by a factor between 4 and 6. This suggests that a reasonable incremental charge per kilopacket is in the range of 20-30 cents.

2.4 Peak Throughput

Usage of the ARPANET differs from node to node. Generally, one can expect two kinds of users in the network --those whose peak bandwidth requirements are not very different from their average bandwidth needs and those who occasionally require very high bandwidths in relation to their average usage. The latter case includes users employing interactive graphics.

The ARPANET as presently configured can allow a typical user to enter or receive transmissions at a peak rate of about 80 KBPS if the network is not heavily loaded. Some users may never require such capacities. Two questions thus arise:

1) Can service to low peak throughput users be provided at lower costs than presently obtainable in the ARPANET?
2) What is the cost to provide peak throughput capability as a function of the number of users who require this service?

In Reports 3 and 4, these questions are independently addressed. To answer question 1 (see Report 4), networks with average throughput requirements of about 6 KBPS/node

(828 KPackets/Day/Node) and 2-connectivity can be supplied by installing two 9.6 KBPS links at the node requiring only low peak throughput. Since the monthly cost for such a line, $650 plus $0.40/mile, is significantly lower than the $850 plus $5.00/mile for a 50 KBPS line, one might conjecture that considerable savings would result for a node not requiring high peak throughput. To test this hypothesis, the following experiment was performed.

A 40 site system was studied and a low cost network was derived using only 50 KBPS lines. The 40 node network had a cost of $1,025,000 and a throughput of 6.0 KBPS/node. Then, five sets of twenty nodes each were randomly selected from among the forty nodes. Each node in each set of twenty nodes was assumed to require low peak bandwidths so that these nodes could be connected into the net by either 9.6 KBPS or 50 KBPS lines, whichever was more economical. The network structure was separately optimized for each set of twenty nodes and the cost savings achieved by allowing the 9.6 KBPS lines was calculated. Finally, all forty nodes were assumed to require only low peak throughputs and the network optimization was repeated.

The results of the experiments dramatically indicated that the 9.6 KBPS line option is not generally useful for the ARPANET. In the vast majority of cases, even when the 9.6 KBPS lines were allowed, the optimization programs selected 50 KBPS lines for the most economical configuration. In fact, in only three cases were 9.6 KBPS lines found useful.

The yearly network line costs for the various optimizations ranged from a low of $973,740 to $1,025,510. The maximum savings $51,770, which would have to be averaged over 20 nodes, represents only 10 percent of the line cost per node which itself is only approximately half the overall cost per node. Furthermore, the average savings for the entire $1,025,000 network is less than $25,000--a very small savings in return for a loss of high peak capacity for half the network.

The strong conclusion is that except in a few special cases (such as connecting a low requirement Seattle node), it is undesirable to use low speed lines in the ARPA Network and thus the ability for users to generate traffic into the network at high peak rates is achieved at essentially no cost.

Question 2 was addressed in Report 3 before the above result was available.

The object of the study in the report is to determine the cost of supplying a few nodes with the capability of inputting traffic at an 80 KB/sec. rate while all other nodes are limited to a 10-15 KB/sec. rate. An initial average base traffic level of 10 KB/Sec/Node (138 KPackets/Day/Node) is assumed.

The method of cost computation operates as follows. The costs for 20,40,60,80, and 100 node networks for the 10 KB/Sec/Node's traffic level are found. It is assumed that an arbitrary node wishes to transmit 80 KB/Sec. for short durations of time. If both the origin and the destination of this traffic is known, the best approach is to make a specific optimization run with this data to determine the cost of adding sufficient capacity to the network so that other users will not be affected.

To obtain guidelines for general cost analysis when both sender and receiver are unknown, we follow the steps below.

1. Compute the effective traffic load per node within the network when a single IMP is adding traffic at an 80 KB/Sec. rate to the net and all other IMPs are generating at a 10 KB/Sec. rate.

2. Determine the cost to construct a network to accommodate the effective traffic load with the specified delay constraint. This computation is based on the assumption that the load is uniformly distributed among all nodes. Let this cost be indicated by ΔGLOB.

3. Determine the average cost of increasing the output capacity of a single IMP to 80 KB/Sec. from about 10-15 KB/Sec. This cost in Report 3 is assumed to be the cost of upgrading two average length lines from 10 KB/Sec. capacities to 50 KB/Sec. capacities. (This cost is ΔLOC = $2 [(350 - 650) + (5.00 - 0.40) Avg. Length] per month). We now know from the results of Report 4 that for overall economy we wish 50 KBPS lines anyway and thus ΔLOC = 0.

4. The average incremental cost to enable a subscriber to input traffic at the 80 KB/Sec. rate is then

$$\Delta \text{ Cost} = \Delta LOC + \Delta GLOB/NHT$$

where NHT is the r[illegible] f nodes which desire high throughput capability.

Note: The equation for ΔCOST assumes that either (1)high input rates occur infrequently or (2) only a few nodes will have high injection rates. With either of these assumptions, it will be relatively unlikely that more than one user will be

generating 80 KB/Sec. at the same time. Thus, the network need only be upgraded to handle one high input rate at a time, and the cost of this capacity increase can be shared by the NHT high throughput nodes. (Naturally, if specific requirements are available, more exact provisions can be made.)

As a final point, we note that the cost-throughput curves for 20, 40, 60, 80, and 100 node networks are essentially linear in the 5 KBPS/Node - 15 KBPS/Sec/Node regions. This means that the incremental costs are independent of the base traffic load (which was initially assumed to be 10 KB/Sec/Node) and thus that incremental costs computed are applicable for a range of traffic loadings.

By combining the results from the two reports, one can conclude:

1) The network cost of providing 80 KBPS peak throughput capability in a network of 40 nodes is approximately $100,000 per year if the network were to be fully loaded without the high peak traffic useage.

2) The network cost of providing 80 KBPS peak throughput under the same conditions as 1) above in a 100 node network

is approximately $150,000 per year if the network were to be fully loaded without the high peak traffic useage.

3) In a 40-node network nominally rated at 10 KBPS/Node and operating at less than 85% of full load, 80 KBPS peak throughput between a pair of nodes can be achieved at zero incremental cost.

4) In a 100 node, 10 KBPS/Node (1380 KPackets/Day/Node) network operating at less than 94% of full load, 80 KBPS peak throughput between a pair of nodes can be achieved at zero incremental cost.

3. PROPERTIES OF LARGE DISTRIBUTED PACKET SWITCHED NETWORKS

3.1 Cost and Throughput

The substantial cost advantages of computer-communications is of particular importance in satisfying the rapidly growing communication requirements of the Defense Department in the 1970s. Because this is a new field, much research must be done to uncover the tradeoffs between cost, reliability, throughput, and time delay for large networks. Such information is essential for long range planning. Throughout the project, we have investigated the characteristics of large ARPANET-like systems. The goal has been to exhibit these characterisitcs as a function of the number of nodes in the system. To do this, systematic procedures were established to vary the number of nodes in the networks under consideration and to find low cost configurations for given levels of traffic. The heart of the procedure were the computational procedures used to analyze and design large network structures.

To establish the basis for the results, we first describe the assumptions and approach. The factors to be discussed are as follows:

1) Network Model

2) Cost Factors

3) Selection of Node Locations

4) Selection of Node-to-Node Traffic

Network Model

The message handling tasks at each node in the network are performed by a special purpose Interface Message Processor (IMP) located at each computer center. The centers will be interconnected through the IMPs by fully duplex communication lines. When a message is ready for transmission, it is broken up into a set of packets, each with appropriate header information. Each packet independently makes its way through the network to its destination. When a packet is transmitted between any pair of nodes, the transmitting IMP must receive a positive acknowledgment from the receiving IMP within a given interval of time. If this acknowledgment is not received, the packet will be retransmitted, either over the same or a different channel depending on the network routine doctrine being employed.

A design goal for the system is to achieve a response time of less than 0.5 seconds for short messages. The final network must also be reliable, and it must be able to accommodate variations in traffic flow without significant degradation in

performance. In order to achieve a reasonable level of reliability, the network must be designed so that at least two nodes or links must fail before it becomes disconnected.

Response time T is defined as the average time a message takes to make its way from its origin to its destination. Short messages are considered to correspond to a single packet which may be as long as 1008 bits or as short as a few bits, plus the header. If T_i is the mean delay for a pack packet passing through the i-th link, then

$$T = r^{-1} \sum_{i=1}^{M} f_i \, T_i$$

where r is the total IMP-to-IMP traffic rate, f_i is the average traffic rate in the i-th link, and M is the total number of links. T can be approximated as

$$r^{-1} \sum_{i-1}^{M} \left[\frac{1/\mu'}{C_i - f_i} + \left\{ (1/C_i \mu') - (1/\mu C_i) + d_i + T_{IMP} \right\} \right] f_i + T_{HOST}$$

where

C_i is the capacity of link i

$1/\mu$ is the average short message length

$1/\mu'$ is the average packet length in the system including long messages, RFNM, header, and parity check

d_i is the propagation delay of link i in seconds

T_{IMP} is the IMP processing time for an average packet

T_{HOST} is the average time delay on the IMP-to-HOST line.

A fixed routing procedure is used in the network optimization. This procedure determines simultaneously both the link flows and capacities and is discussed in the section on routing.

Cost Factors

Cost factors used are those currently applicable to the present ARPA network. These factors apply to both nodes and links and vary with the capacities of these elements. The factors used are given in Tables 3.1 and 3.2.

TABLE 3.1
LINE COSTS

Capacity (bits per second)	Fixed Cost/Month $	Cost/Mile/Month $
9 600	650.00	0.40
19 200	850.00	2.50
50 000	850.00	5.00
230 400	1 300.00	30.00

All lines fully duplex

TABLE 3.2
NODE COSTS

Description	Rental Cost/Year $	Maintenance Cost/Year $
DDP-516 IMP with up to 7 fully duplex I/O channels.	25 700	7 600
DDP-316 IMP with up to 5 fully duplex I/O channels. Processing rate is 3/4 that of 516 IMP	12 600	5 000

Selection of Node Locations

Node locations strongly influence network efficiency. For example, at one traffic level a 16-node network may be more efficient than the same network with two additional nodes while it may be less efficient at other traffic levels. Consequently, a systematic procedure is required to generate realistic systems with differing numbers of nodes.

The nodal distribution selected for this study is based on population. A list of the one hundred most populated metropolitan areas is used. A required ratio R of population to number of IMPs is assigned to each node, so that if there are P people in a given metropolitan area, ⟨P/R⟩ nodes are assigned to that area, where ⟨x⟩ represents the largest integer no greater than x. Therefore, varying R varies both the number of nodes in the system and the location of these nodes.

As an example, if R = 2,000,000, only areas with at least 2,000,000 people would be assigned an IMP. Since the New York area has a population of 10,602,382, it would have five IMPs. The Los Angeles area would have three IMPs, the Chicago area three IMPs, and so on. As a conservative figure, distances between IMPs in the same metropolitan area are taken to be twenty miles. The overall system would then contain nine

distinct metropolitan areas with a total of 18 IMPs. Table 3 .3 indicates these relationships as a function of R

TABLE 3.3

NUMBER OF IMPs IN RELATION TO POPULATION

No. of IMPS(N)	R= population/IMP	No. of Cities Used
0	over 10 602 382	0
5	4 301 283	4
10	2 725 841	6
15	2 150 641	8
20	1 871 723	10
25	1 667 244	13
30	1 362 920	14
35	1 247 816	15
40	1 183 514	18
45	1 071 003	20
50	1 023 238	22
60	893 370	26
70	815 568	28
80	718 685	32
90	666 897	36
100	623 670	40

Selection of Node-to-Node Traffic

A fundamental problem in all network design is the estimation of the traffic the network must accommodate. For some problems accurate estimates of user requirements are known. However, complete studies are not yet available to predict the flow requirements in networks of the type being considered here. A number of basic questions are yet to be resolved. For example, it may be reasonable to assume that the flow out of a node will be proportional to the population assigned to that node. However,

will the flow between two nodes be affected by the distance between these nodes? If so, how will the cost-throughput characteristics of the network be affected?

In order to investigate the effect of different traffic distributions on network economy, a sequence of experiments was conducted in which traffic patterns were varied and low cost networks for these patterns generated. The traffic out of each node in city C_i with population P_i is equal to $KP_i/\langle P_i/R\rangle$ where R is the required population per IMP, and K is a constant which determines the traffic level. The traffic from each node in this city to any node in city C_j is equal to

$$K[P_i/\langle P_i/R\rangle] \frac{[P_j/\langle P_j/R\rangle] d_{i,j}^{\alpha}}{\sum_k [P_k/\langle P_k/R\rangle] d_{i,k}^{\alpha}}$$

where $d_{i,j}$ is the distance in miles between C_i and C_j for $i \neq j$, $d_{i,i}$ = 20 miles, and α is a non-negative constant. If $\alpha = 0$ traffic between cities is independent of distance while if $\alpha = 1$, traffic follows the well known gravity law.

For a fixed number (N) of nodes and fixed α, K is varied. Each value of K gives a specified distribution of traffic for which a low cost network can be found. For each K a point on curves of cost versus bits/s/node and cost versus cost per

megabit of transmitted information is thus determined. By varying K, a number of values of these parameters are found. This procedure was applied for N=40 and α = 0.0, 0.5, and 1.0. The results tend to indicate that for given throughputs, only a few percent lower cost designs can be obtained for greater values of α, that is, for traffic patterns which favor flows between nodes close together. However, the cost differentials between traffic patterns with α = 1, the gravity law, and α = 0, no distance bias, do not appear to be significant. Therefore, for later studies traffic patterns without distance bias are used exclusively. Thus, the flow from a node in city C_i with population P_i to a node in city C_j with population P_j is assumed to be

$$K[P_i/\langle P_i/R\rangle] \frac{[P_j/\langle P_j/R\rangle]}{\sum P_k/(P_k/R)}$$

for $i \neq j$. The flow from a node in city C_i to another node in the same city is

$$\frac{KP_i^2/\langle P_i/R\rangle^2}{\sum P_k/\langle P_k/R\rangle}$$

Naturally, there is zero flow from any node to itself.

Large Network Cost-Throughput Characteristics

In this section we indicate cost-throughput tradeoffs as a function of the number of nodes in the network. The following is a summary of the factors which influence the network designs.

1) Systems considered contain 20, 40, 60, 80, 100 and 200 IMPs and cover respectively the 10, 18, 26, 32, 40 and 62 largest metropolitan areas in the Continental United States.

2) Required traffic between any two IMPs is independent of distance. From an IMP in city C_i with population P_i to an IMP in city C_j with population P_j, the traffic flow requirement is

$$\frac{K[P_i/\langle P_i/R\rangle]\ [P_j/\langle P_j/R\rangle]}{\sum_k [P_k/\langle P_k/R\rangle]}$$

where K is a positive constant and R is the required population per IMP.

3) Messages are assumed to have the same packet structure and formats as in the ARPANET and 85% of all messages are assumed to be single packets.

4) In any acceptable network design, a minimum of two nodes or links must fail before all paths are broken between any pair of nodes.

5) Throughput is equal to the average number of bits/second/node which causes an average short message response time of 0.5 seconds.

6) Only hardware presently being used in the ARPANET is used in any design. Only communication link options at present available are used in the design.

An important point which must be emphasized is that the results to follow present a conservative picture of the relationship between cost and throughput. There are two major reasons for this.

1) Each point represents a feasible network obtained by the computer network design program. Thus, to generate the specified throughput, no greater cost would be involved. However, because of the number of points needed to generate adequate curves, it is prohibitively costly to devote a large amount of computer time to optimize completely each design point. Therefore, if a specific throughput were to be required, a more intensive optimization would be warranted and a lower cost design would be probable.

2) In each design, only hardware and line options at present available have been allowed. Other equipment is being developed and other communication options will be available in

the near future. For example, we discussed in Report 1 the economies created by using a 108 kilobit/second data set. Although this data set has been developed by AT&T, it is not yet a commercial offering. However, the costs involved in building a large computer network would justify the independent development of such a data set. Moreover, higher rate IMPs are now being developed with existing hardware. Such IMPs might considerably enhance network performance at high data rates.

Because of the above factors, the numerical relationships that follow can be considered to be the result of a worst case analysis. However, we feel that they represent realistically the behavior of the networks under consideration and that while some reductions in cost or increases in throughput are possible, the fundamental relationship between these quantities is accurately depicted.

Figures 3.1, 3.2 and 3.3 show cost, throughputs, and other relevant data for a number of 20, 40, 60, 80, 100, and 200-node networks. Figure 3.1 indicates the relationship between cost per node and the average number of bits out of each node. In the figure, the location of a point corresponding to an i x 10-node network is indicated by the numeral i. In a similar manner, Figure 3.2 shows the

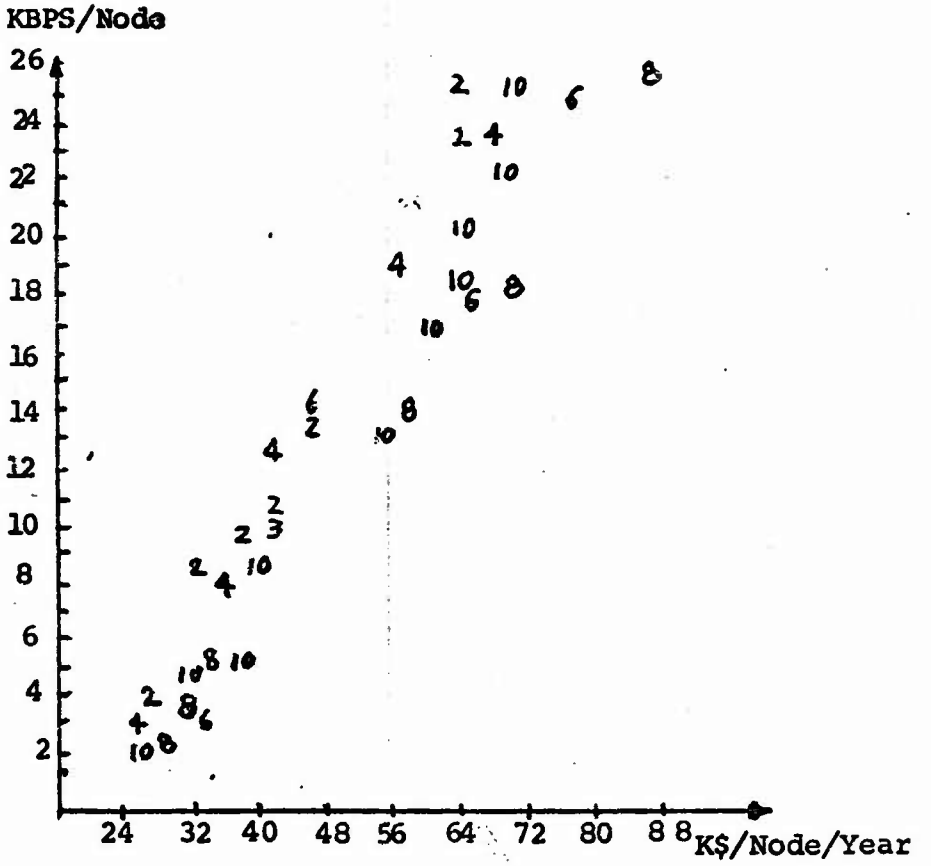
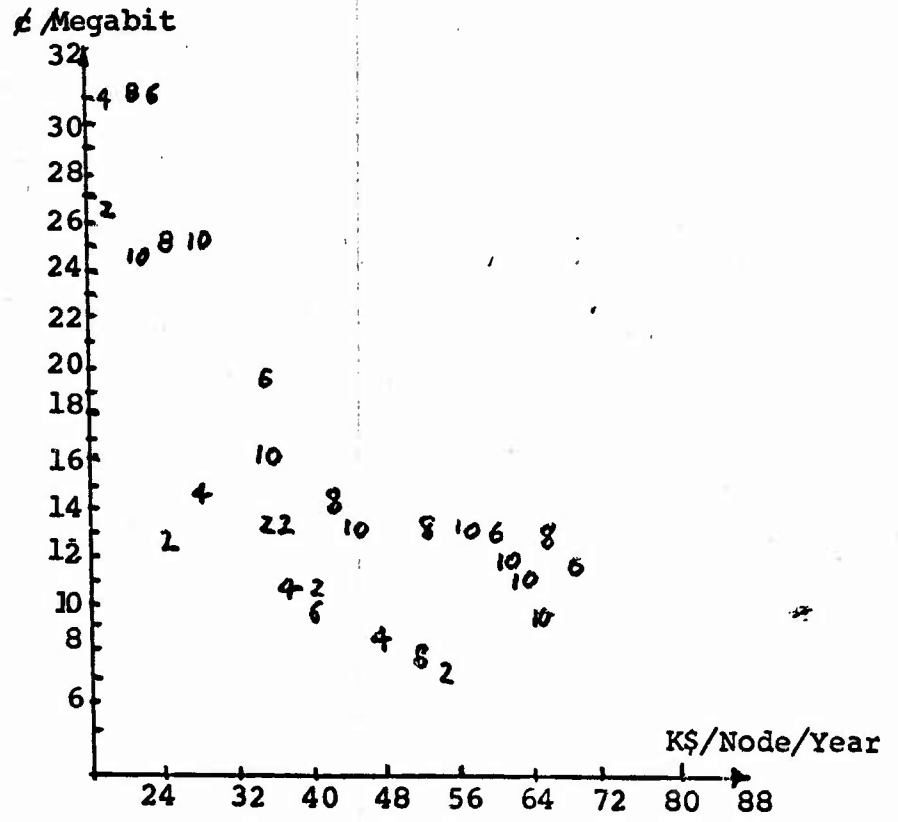


FIGURE 3.1 NODE COST VERSUS NODE THROUGHPUT



COST PER MEGABIT IN NETWORK VERSUS COST PER NODE

FIGURE 3.2

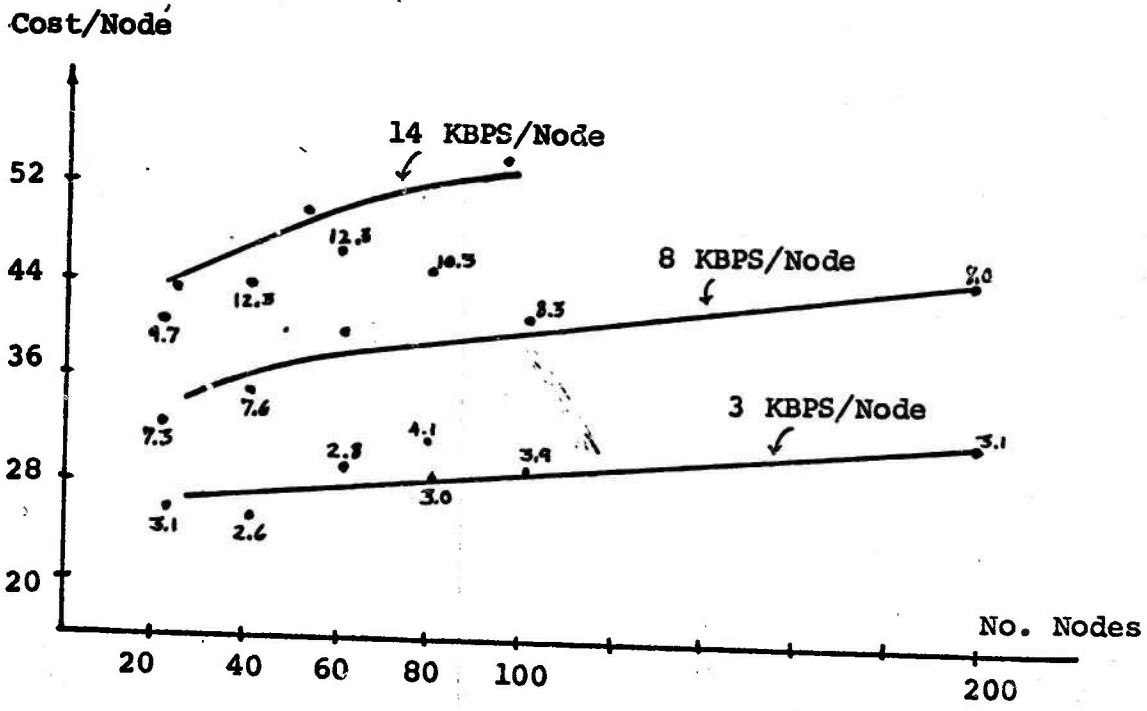


FIGURE 3.3

COST PER NODE VERSUS SIZE

tradeoffs between cost per node and cost per megabit of transmitted data.

In Fig. 3.3, the average number of bits out of each node is taken as a parameter. In this figure, the abscissa corresponds to the number of nodes in the network while the ordinate represents the cost per node. The number attached to each point is the average output per node for that network.

Fig. 3.3 clearly indicates trends in network performance and economy. At low throughputs, 3000 bits/s or less, within the range of the sizes considered, the larger networks are more economical than the smaller networks. This is because there is a minimum cost required to construct a two-connected network, independent of throughput. The minimum cost networks for 20 and 40-node systems result in throughputs of approximately 3000 bits/s and so if less than that throughput is required, the full cost is still applicable. The larger networks introduce economies of scale which allow more efficient sharing of capacity. Thus, a throughput per node of, say, 3000 bits/s per node can be obtained more economically for a 80-node network than for a 60-node network.

At moderate and high output levels, 5000 to 15,000 bits/s per node, the larger networks appear to be nearly as economical

as the smaller networks when communication costs are compared. However, since any network would require certain overall fixed project and management costs, operating with a larger number of nodes would reduce the total cost per node for the system on a per node basis beneath that required by small networks.

3.2. Reliability

Small to Medium Networks (NN≤50)

The initial design procedure for the ARPANET (Report 1) controlled reliability by insisting that there be at least two node disjoint paths between every pair of nodes. Such networks are said to be two-connected. Later computations (Report 3) proved that this implied almost perfect reliability in the following sense. Suppose node i in the network is inoperative a fraction p_i of the time for i=1, ..., NN. Then a lower bound for the expected number of node pairs which cannot communicate is equal to the expected number of node pairs not communicating in a complete network where each node pair is joined by an invulnerable link. No addition or redistribution of links can reduce the expected number of node pairs not communicating below this value. For small networks, the existence of two node disjoint paths between each pair of nodes invariably resulted in an expected number of node pairs not communicating very near the lower bound. Thus, the addition of more links for reliability purposes was not justified. The calculation of this important lower bound is as follows:

Let each node i of a network with NN nodes have a probability p_i of failing. Then, the expected number of node pairs in which one or both nodes have failed is

$$\sum \{1-(1-p_i)(1-p_j)\}$$

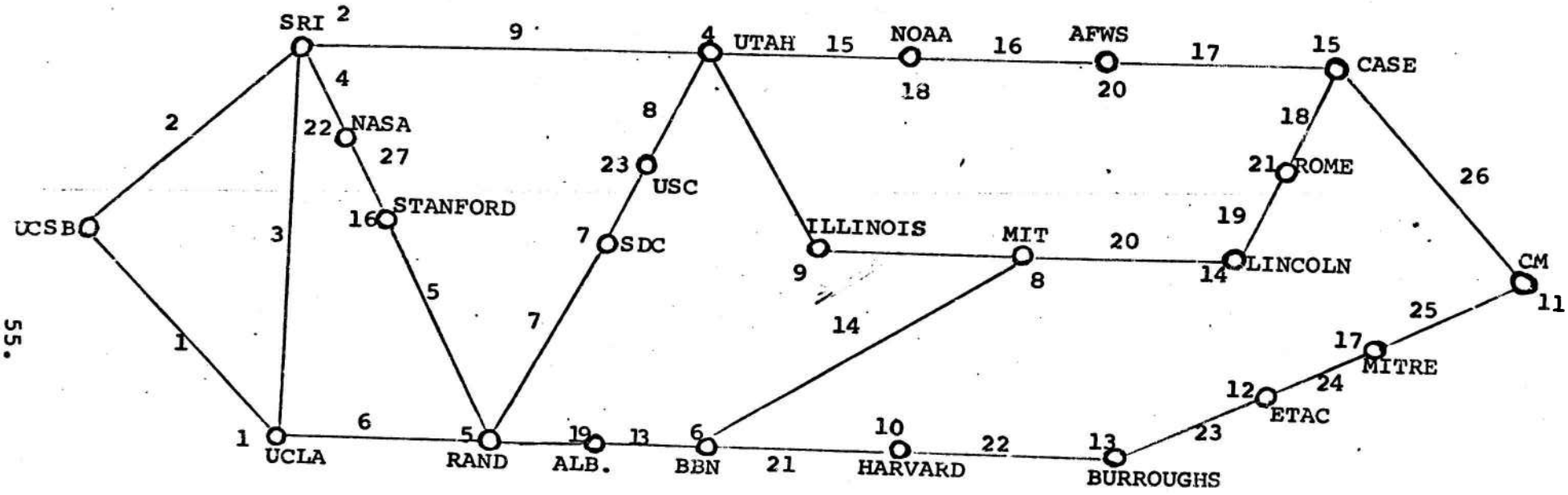
If $p_i = p$ for i=1, ..., NN, i < j then the expected number is

$$NN(NN-1)\left\{1-(1-p)^2\right\} = NN(NN-1)[2p(1-p)]$$

and the expected fraction of node pairs with at least one node failed is [2p(1-p)]. Two important implications of this simple result deserve to be emphasized. First, the expected fraction of non-communicating node pairs cannot be reduced below [2p(1-p)], and second this lower bound is invariant with respect to the size of the network.

To fix these ideas and to give specific examples of the reliability characteristics of small networks, we consider two versions of the ARPANET. The first is a 23-node network that has been thoroughly analyzed (Report 3) as a common measuring point or standard for the various reliability analysis techniques. The second network is a medium size network of 33 nodes in which for the first time an additional link was considered mainly for reliability reasons. The 23 node network is represented in Figure 3.4.

23 Node Network



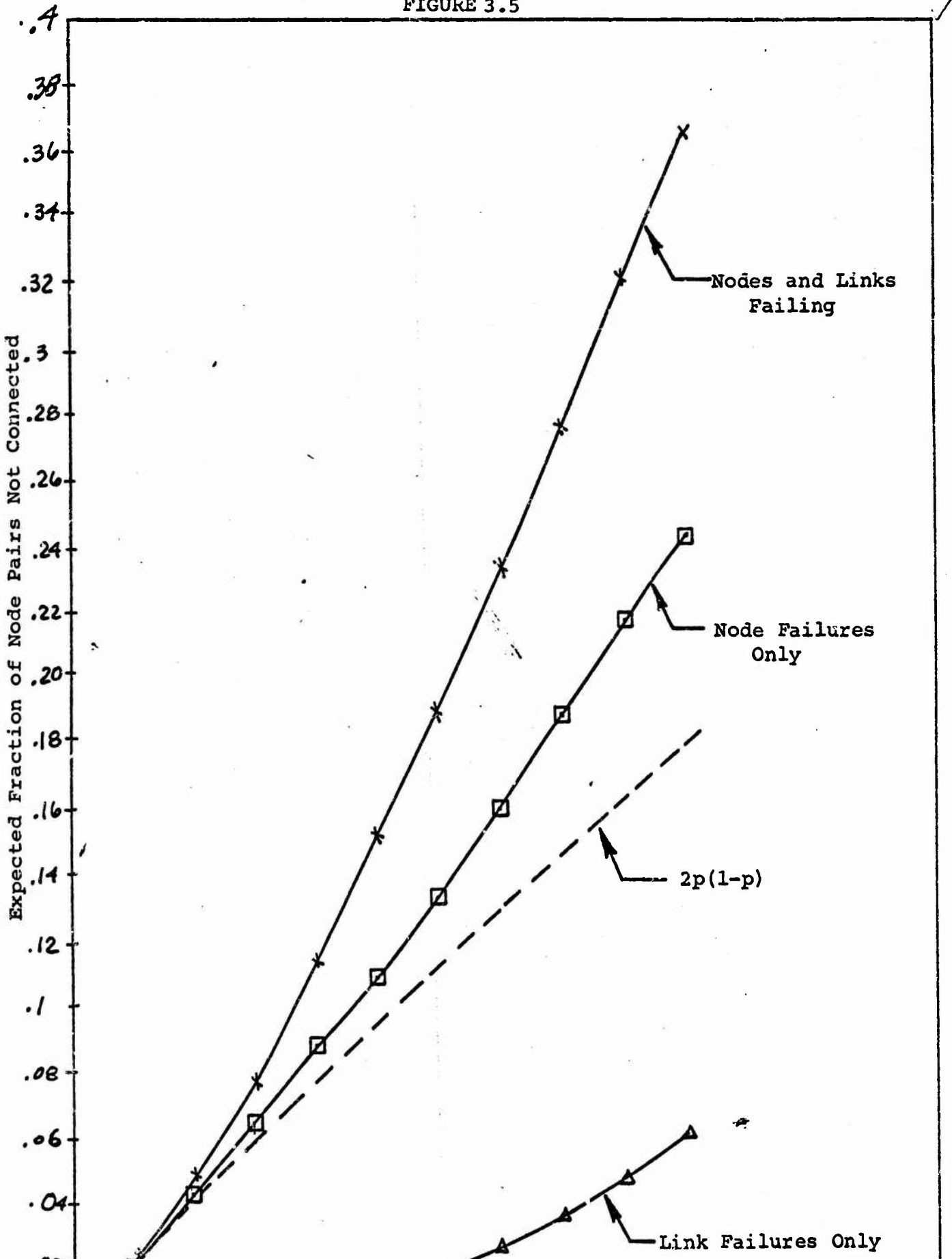
Base Network for Reliability Analysis

FIGURE 3.4

The measured average downtime on the ARPANET links is approximately 2% and hence we assume a base element failure probability of 0.02. Then, $2p(1-p)$ equals 0.0396 for $p = .02$ and hence the expected fraction of node pairs not communicating must be at least $(.0396)(23)(22)/2$ equals 10.0188. In Figure 3.5 the expected fraction of node pairs not communicating as a function of element failure probability is shown. Also shown is the expected fraction of node pairs not communicating when only links fail, when only nodes fail and finally when the curve $2p(1-p)$ is plotted. For $p = .02$, the expected fraction of node pairs not communicating is 0.049.

In the case where only nodes fail the expected fraction is .0427 and for only links failing.0018. Remembering that $2p(1-p) = .0396$, we see that 80% of the node pairs which cannot communicate can be ascribed to purely the fact that one of the nodes of the pair in question has failed. Thus, the improvement in relability to be gained by changing the network configuration is minor. The expected fraction of non-communicating node pairs does not completely reflect the degradation of throughput due to element failures. The most detailed level of analysis of reliability incorporates element failures, flow requirements, routing, acceptable delays and other pertinent network characteristics. In order to test the adequacy of the ARPANET under

ANALYSIS OF NETWORK RELIABILITY
FIGURE 3.5



the most stringent of conditions, a reliability analysis treating these factors was performed (Report 4). The effect on throughput at average delay of 0.2 seconds was examined by removing nodes and links from the network and applying the routing and analysis algorithms to the remaining network. The nominal throughput of the 23 node network with all elements operable is 11.5 KBPS/node. When nodes and links are failing with p =.02, the expected throughput is at least 9.0 KBPS/node. These results again show that for small networks, reliability is not a dominant factor.

Fig. 3.6 shows a 33 node network. For the network without the link (shown dashed) between FT.BEL and ABER, the difference between the expected fraction of node pairs not communicating =.058 and 2p(1-p) = .040 is almost double the difference for the 23 node network so that improving the reliability by changing the network configuration becomes marginally feasible. An extra link from FT.BEL to ABER increased the cost by a little over 1% and increased the reliability by almost 10%. Thus, even for a network with only 33 nodes, it is valuable to consider reliability in more detail than the "two connectivity" criterion. For failure rates substantially greater than 2% it is even more important.

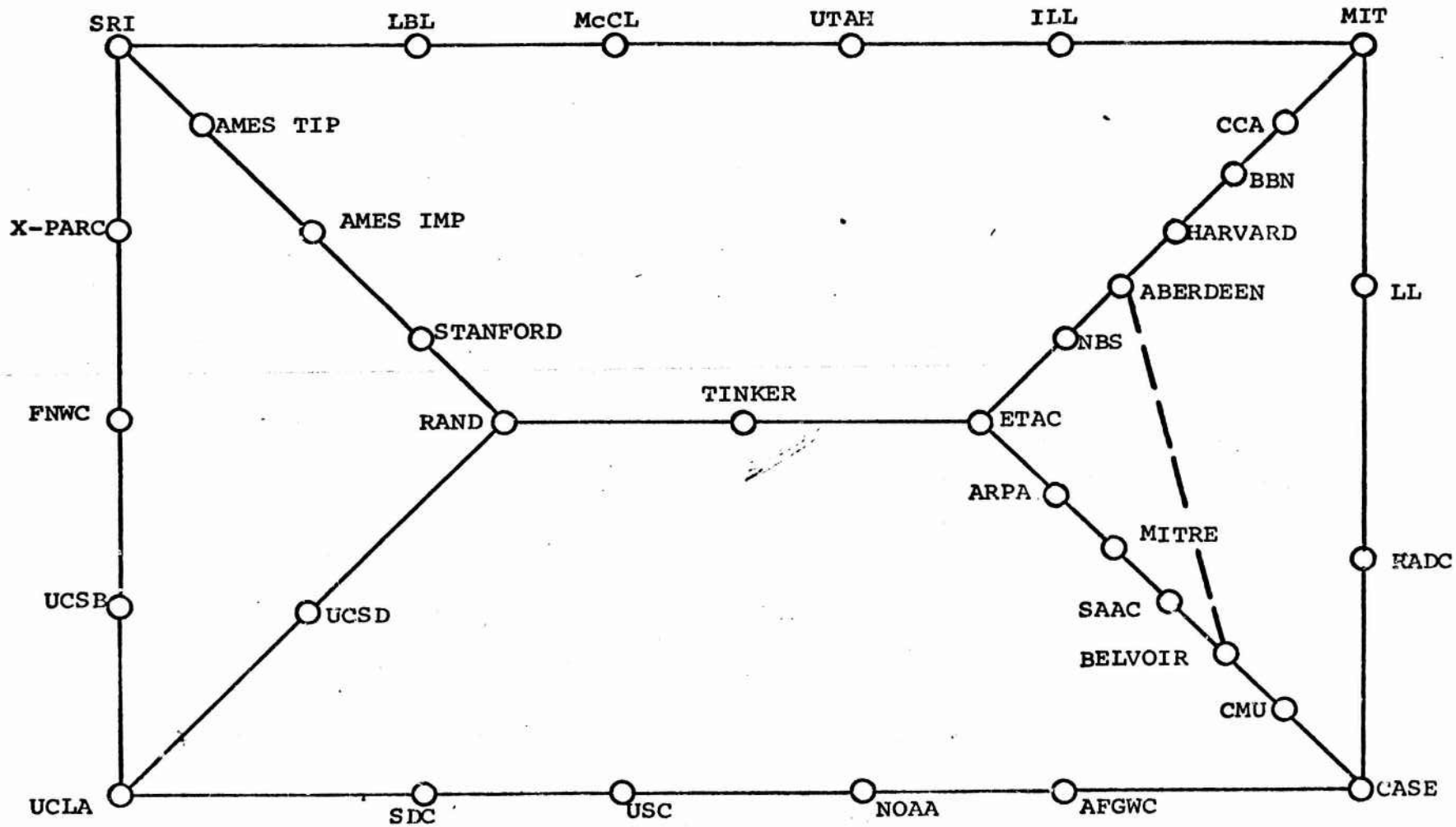

SRI
LBL
McCL
UTAH
ILL
MIT
AMES TIP
CCA
BBN
X-PARC
AMES IMP
HARVARD
ABERDEEN
LL
STANFORD
NBS
TINKER
FNWC
RAND
ETAC
ARPA
MITRE
RADC
UCSB
UCSD
SAAC
BELVOIR
CMU
UCLA
SDC
USC
NOAA
AFGWC
CASE

FIGURE 3.6

Reliability Trends for Large Networks

While for smaller networks and low element failure probabilities ($p \leq .02$), it was found that designing the network with at least two node disjoint paths between each node pair for throughput in the range 0-15 kilobits/second/node guaranteed sufficient reliability, as networks become larger this simple approach fails. The first experiments which indicated this started with the low cost networks of 20, 40, 60, 80, 100 and 200 nodes with throughput approximately 8 KBPS/node discussed earlier and designed with the reliability constraint of two node disjoint paths. The results are shown in Fig. 3.7 when nodes are perfectly reliable. As measured by the fraction of node pairs not communicating, the reliability actually increased with the number of nodes up to 60 nodes at which point the reliability began to decrease. As is evident, the decrease in reliability is dramatic even though nodes have been assumed to be perfect.

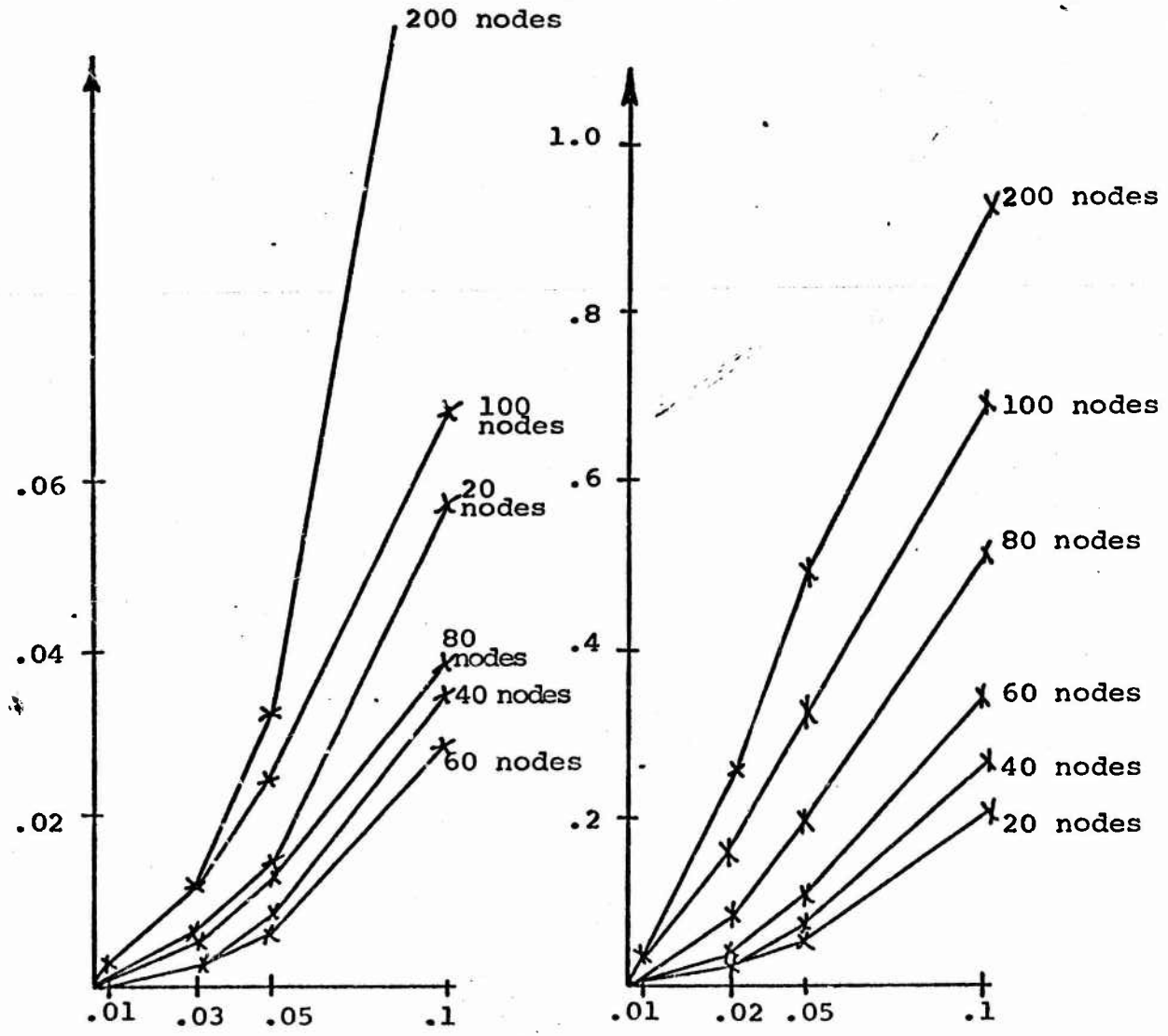
Fig. 3.8 shows the results of analysis of a family of two and three connected* networks containing from 20 nodes to 200 nodes. The networks analyzed contain 20, 40, 60, 80, 100 and 200 nodes. However, a continuous line is drawn for visual

*A k-connected network has k node disjoint paths between every pair of nodes.

FIGURE 3.7
NETWORK RELIABILITY AS A FUNCTION OF NUMBER OF NODES

Fraction of node pairs not communicating versus probability of link failure

Probability of Network being disconnected versus probability of Link Failing



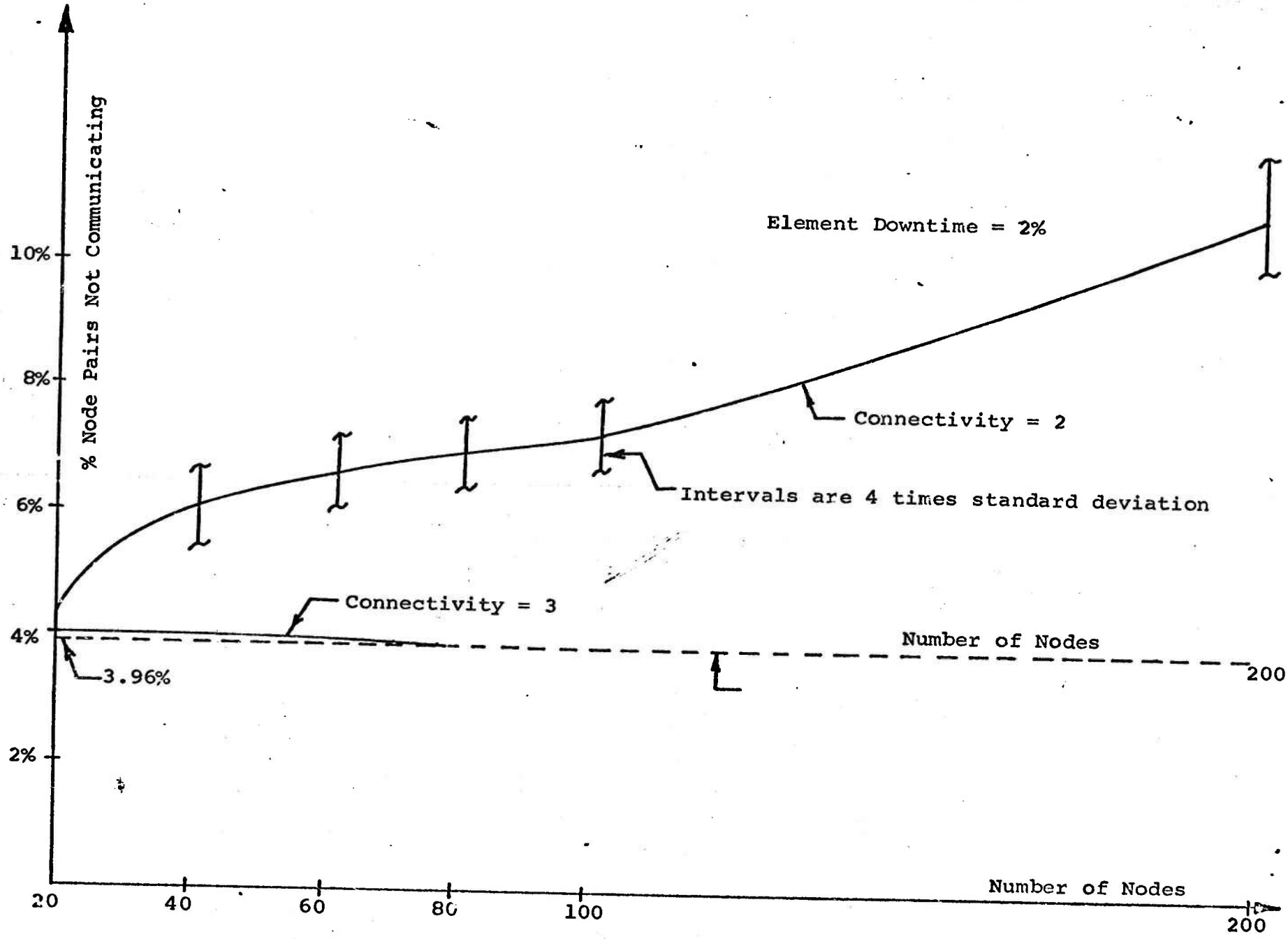

% Node Pairs Not Communicating
10%
8%
6%
4%
2%
Element Downtime = 2%
Connectivity = 2
Intervals are 4 times standard deviation
Connectivity = 3
Number of Nodes
200
3.96%
20
40
60
80
100
Number of Nodes
200

FIGURE 3.8

convenience. It can be seen from Fig. 3.8 that when there are 3 node disjoint paths between every pair of nodes, for p = 0.02, the unreliability is close to the ideal minimum which results from only the node failures. Similar conclusions for p = 0.05 and 0.10 hold (see Report 5). From these, we can conclude that requiring 3 node disjoint paths between every pair of nodes is sufficient to essentially guarantee an optimal reliability with respect to link allocations for networks with less than 200 nodes and for element failure probabilities of less than 0.1. As is evident, this results in a potential overdesign of the system but it clearly demonstrates the economic viability of distributed network designs with near perfect reliability.

Very Large Networks

Exact analyses for very large distributed networks containing a thousand or more nodes have not yet been performed. (For "tree"type networks, such analyses can easily be performed using the techniques of Report 5). However, some preliminary conclusions based upon approximate analyses of sample structures and the use of theoretical bounds are possible.

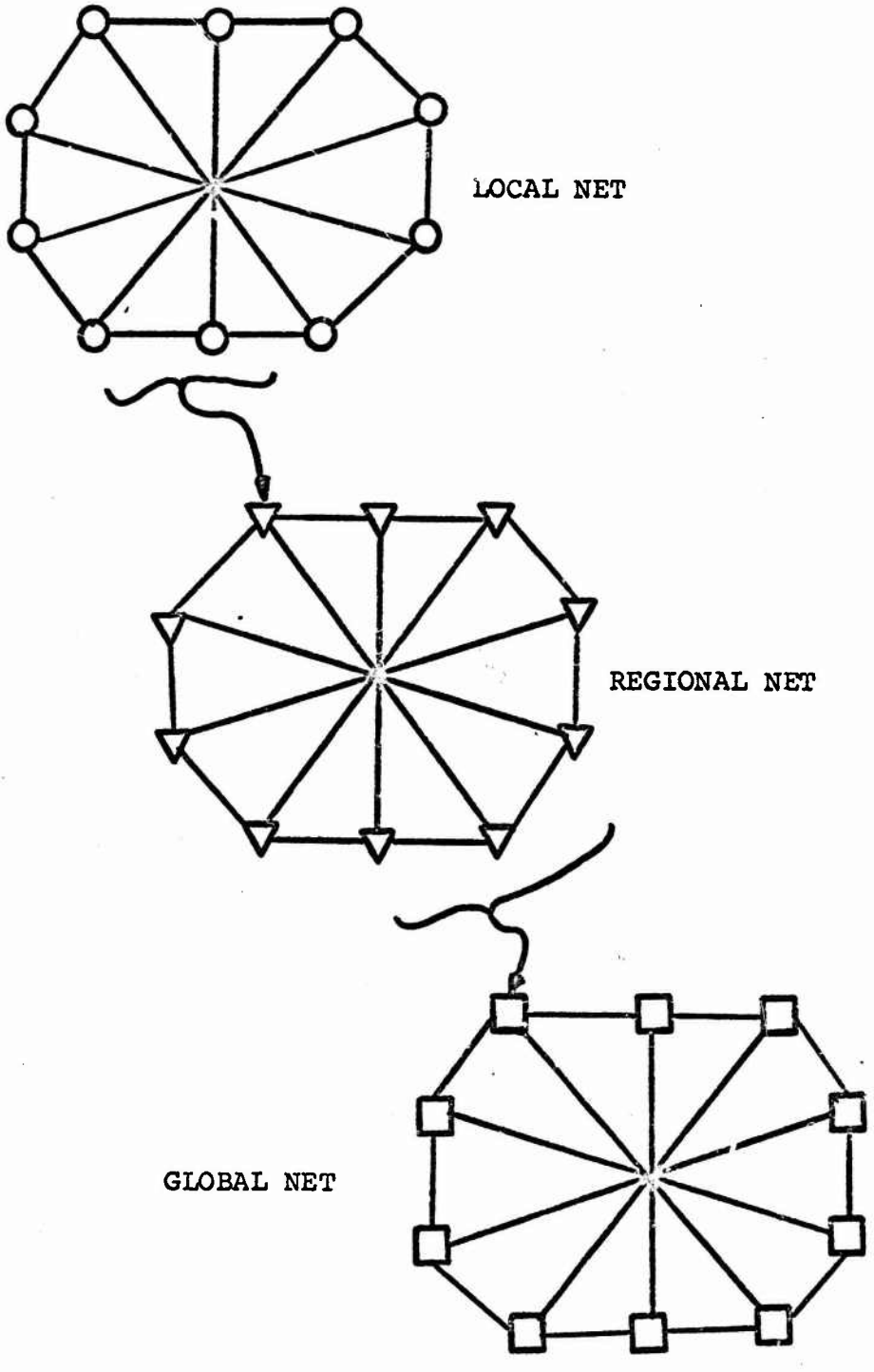
1) Most nodes will require near perfect reliability such as might be achieved by installing redundant message processors

at every node or by using the new high reliability IMPs now under development.

2) Two connectivity, even with perfect node reliability, will provide inadequate reliability when present link downtimes are considered. For example, a theoretical upper bound indicates that a 2-connected 1000 node network cannot have more than a .68 probability of connectivity.

3) If redundant message processors are used, high reliability can be achieved by connecting 3-connected subnetworks together. For example, the 1000 node network shown in Fig. 3.8 has a .973 probability of connectivity.

Consequently, we can conclude that from a reliability standpoint, a 1000 node ARPANET would be feasible using the new IMPs (or the original first generation IMPs in a redundant configuration) and no more than a 3-connected network structure such as the one shown in Fig. 3.9 . Such a network would contain no more than 1665 links, of which 1500 would be involved in local communications, 150 in regional communications and 15 in nationwide communications. This means that the prime candidate for decreasing communication line cost would be the development of low cost/high reliability local distribution schemes.



A 1000 node network composed of 10 ten-node regional nets each containing 10 ten node local nets.
FIGURE 3.9

4. COMPUTATIONAL TECHNIQUES FOR ANALYSIS AND DESIGN

Network analysis and design techniques generally incorporate a set of basic procedures for finding the components of a network, generating minimum spanning trees on a set of nodes, and generating shortest paths in a network. These problems are encountered in such diverse areas as least cost electrical wiring, mininum cost connecting communication and transportation networks, network reliability problems, clustering and numerical taxonomy, algorithms for solving traveling salesman problems, and multiterminal network flows. Furthermore, in optimizing the design of a network, one is often required to apply analysis algorithms which generate these structures many thousands of times. Therefore, it is essential to carry out these computations as efficiently as possible.

In this chapter, we describe improved algorithms for the above network problems as well as algorithms for routing and reliability analyses in computer communication networks. The new procedures take advantage of special network structures, data representations, and fundamental network properties to extend the sizes of networks that can be tractably studied.

4.1. Finding Components of a Network

An important basic network problem is the formulation of an efficient method to determine whether an undirected network is connected. For example, for a single value of link failure probability thousands of connectivity calculations must be carried out in a Monte Carlo simulation for network reliability. Given the network in node adjacency form, a very efficient method of determining the components is the following:

Algorithm A:

Step 0 (Initialization): Set i=1, j=1, S=$\emptyset$. Label node i=1 with component label j=1.

Step 1 (Look at new link): Find the next node i' adjacent to i; if there are none, go to Step 3. If the node i' is not already in a component, go to Step 2. If node i' is already labeled with a component number, repeat Step 1.

Step 2 (Add a node to current component): Label the node i' with the current component label j and add the index of the labeled node to the stack S. Return to Step 1.

Step 3 (Scan a new node): Remove a node index i" from S and set i equal to i". Go to Step 1. If S is empty go to Step 4.

Step 4 (Current component complete--start a new one): Set k to k+1. If k>l, we're done; otherwise if node k is unlabeled, set i to equal k and set j to j+1. Go to Step 1. If node k is labeled, repeat Step 4.

This algorithm terminates with each component having a different label. If the links (i,i') occuring in Step 2 are saved, one also obtains a spanning forest. The order of computation is linear in n and m although if the network is nearly complete, the number of links m is quadratic in n. If one is interested only in determining if the network is connected or not, the algorithm can be terminated the first time Step 4 is encountered. This algorithm is probably close to being optimally efficient if the links are given in node adjacency form.

Algorithm A has the disadvantages that the links incident to a node must all be scanned before links incident to other nodes can be worked on. This is necessary in order to avoid relabeling nodes. For example, this restriction prevents one from adding, in a simple way, links to a net already analyzed. A slightly slower but much more flexible algorithm is:

Algorithm B:

Step 0 (Initialization): Start with $A_0 = \emptyset$ and assign each node a separate component label. Set k=0 and go to Step 1.

Step 1 (new link): Add a link $a_k = (i_k, j_k)$ to A_k to form A_{k+1} (if there are no remaining links, i.e., A_k=A stop). Examine the component labels of i_k and j_k; if they are the same, repeat Step 1 with k set to k+1. If not, go to Step 2.

Step 2 (Join components): Change all the node labels which are the same as the label of i_k (including i_k's label) to the label of j_k. Set k to k+1 and go to Step 1.

The order of computation is dominated by the relabeling in Step 2 which occurs n-c times where c is the number of components. Using a straightforward implementation, each time through Step 2 the labels on all n nodes have to be checked in order to relabel. Thus, on the order of n^2 operations are involved with relabeling.

In the algorithm developed in Report 4 a list structure was maintained so that only nodes for which the labels are changed are considered. Further, the number of nodes in each component was maintained so that it was possible to change the labels on the smaller of the two components joined in Step 2. This reduces the maximum order of computation to $m \log_2 n$ plus a term linear in m. This increase in speed by using list structures does incur an expense in storage requirements.

4.2. Minimum Spanning Trees (Shortest Trees)

Until recently, the most efficient methods for calculating minimum spanning trees (MST) or forests was to use Prim's algorithm for nearly complete graphs which involves on the order of n^2 calculations or using the Kruskal Algorithm with Treesort (see Appendix A of Report5) and Algorithm B. The sorting pass takes on the order of $m\log_2 m$ calculations while the second pass involves $n\log_2 n$ dependence on the number of nodes n and depends linearly on the number of links. Thus, for sparse networks where $m\log_2 m$ is small compared to n^2, the modified Kruskal algorithm will be faster.

The first general improvement is to notice that the main expense in the modified Kruskal Algorithm is in the sorting which takes in general $m\log_2 m$ operations and to notice that most of the links are not considered because they make cycles with shorter links. In Treesort (Appendix of Report 5) applied to the list of link lengths, the list is first arranged into a binary tree which is a "heap"; that is, each link length is no longer than its descendants in the tree. This takes about m interchanges and 2m comparisons at the worst. Next, the top link corresponding to the top of the heap is considered via Step 1 of Algorithm B for the MST. Then the link length

is deleted from the heap and a new link length corresponding to link (i,j), say, is taken from the bottom to the top and the heap restored by a sift-up. The sift-up takes at most $2\log_2 m-1$ interchanges and at most $2\log_2 m-2$ comparisons to restore the heap. Often the sift-up can be saved by comparing the component labels of i and j in Algorithm B. If they are the same, the link forms a cycle with a shorter link and can be discarded immediately. Using this approach, the sorting cost is on the order of $2m+k\log_2 m$ where k is the number of links examined in Algorithm B before a spanning tree is obtained since only the links actually considered for the MST are sorted. In general, one may have to examine all m links but for nearly complete networks this is unlikely. In Report 5, it is shown that using this further modification to Kruskal's Algorithm, it becomes nearly as efficient as Prim's Algorithm for complete graphs and is still much better than Prim's Algorithm for sparse graphs.

However, Prim's method can also be improved. We use Treesort to determine $d_j=\underset{i}{\text{Min}}\, d_i$ in Step 1 of Prim's Algorithm. We assume the d_i form a "heap". The top of the heap is d_j* which is then removed from the heap. Then, in Step 2 of Prim's Algorithm some of the d_j become smaller and are modified.

Next, a d_j from the bottom of the heap is moved to the top. Finally, the heap is restored. At the worst, each restoration of the heap takes a number of operations linear in n and usually considerably less is needed especially if the network is sparse. Even if it is not sparse, many of the d_j do not change; furthermore, all but one of the d_j which change decrease in value so that the "sift-up procedure" of the Treesort takes on a particularly simple form.

Numerical experiments were carried out on randomly generated networks. For given n and m, random networks of m links and n nodes were generated. Then, a random length between 0 and 1 were generated for each link. The distribution of lengths is of no importance since any equivalent pre-ordering would give the same results; thus, any method which generates random permutations of the links will suffice. Three series, four networks each, were used. These are given in Table 4.1.

TABLE 4.1
NETWORK SERIES USED

Series		(n,m)
1	m=n-1	(10,9) (50,49) (100,99) (500,499)
2	m=3n	(10,30) (50,150) (100,300) (200,600)
3	m=n(n-1)/2	(5,10) (10,45) (20,190) (40,780)

One hundred samples of each of the 12 network sizes were analyzed by each of four algorithms: Prim's Algorithm, Modified Kruskal, Prim's Algorithm with sorting, Modified Kruskal's with partial sorting. Each algorithm was presented with exactly the same networks. For each of the algorithms and each of the network sizes, the analysis time for each of a hundred trials was obtained. The maximum over 100 trials, the average over the 100 trials and the standard deviation over the hundred trials were recorded. The computer clock gives results in milliseconds and the clock routine itself takes less than one half millisecond. The results are presented in tabular form in Table 4.2 and in graphical form in Figure 4.1. As is suggested by theory, Prim's Algorithm works best for the complete graphs and Kruskal's Algorithm works better for sparse graphs. The two algorithms using sophisticated versions of Treesort yield good results over a wider range of sparsity. The modified Kruskal Algorithm with partial sorting is apparently the best if speed over a wide range of sparseness is the criterion.

TABLE 4.2

COMPARISON OF RUNNING TIME AVERAGE OVER 100 TRIALS FOR FOUR MST ALGORITHMS

n	m	Kruskal	Kruskal/ Partial Sort	Prim	Prim Sort
5	10	2.2	1.7	1.59	2.24
10	45	10.2	5.64	4.64	6.43
20	190	51.6	19.6	16.07	20.13
40	780	263.95	67.79	58.48	63.24
10	9	2.29	2.34	3.36	3.91
50	49	14.52	14.43	56.70	30.68
100	99	32.28	31.81	206.75	74.27
500	499	193.98	195.33	*	536.76
10	30	6.72	4.71	4.27	5.81
50	150	42.44	35.97	59.03	45.46
100	300	93.41	85.30	212.29	110.98
200	600	206.22	199.49	804.40	245.61

*(n=500, m=499 not computed for Prim)
In milliseconds on a CDC 6600 Computer.

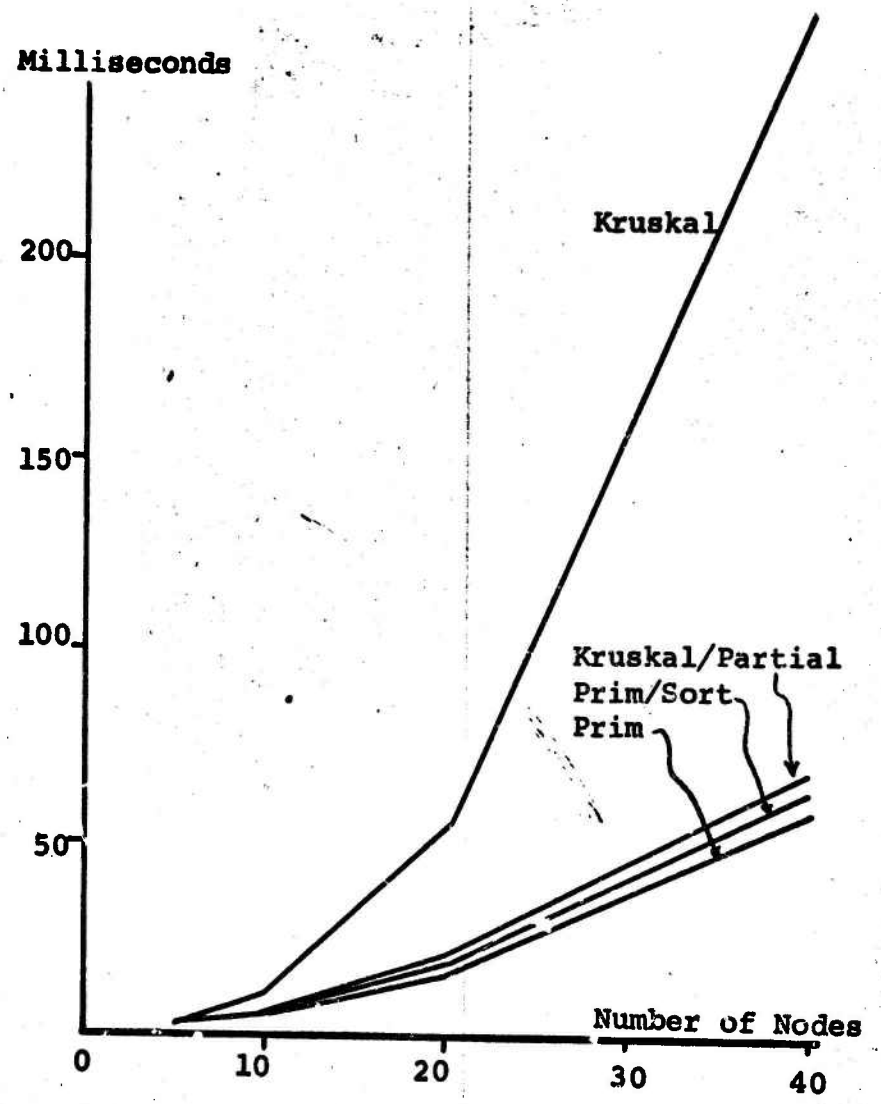


FIGURE 4.1(a)

TIME IN MILLISECONDS TO FIND MST
m=n(n-1)/2

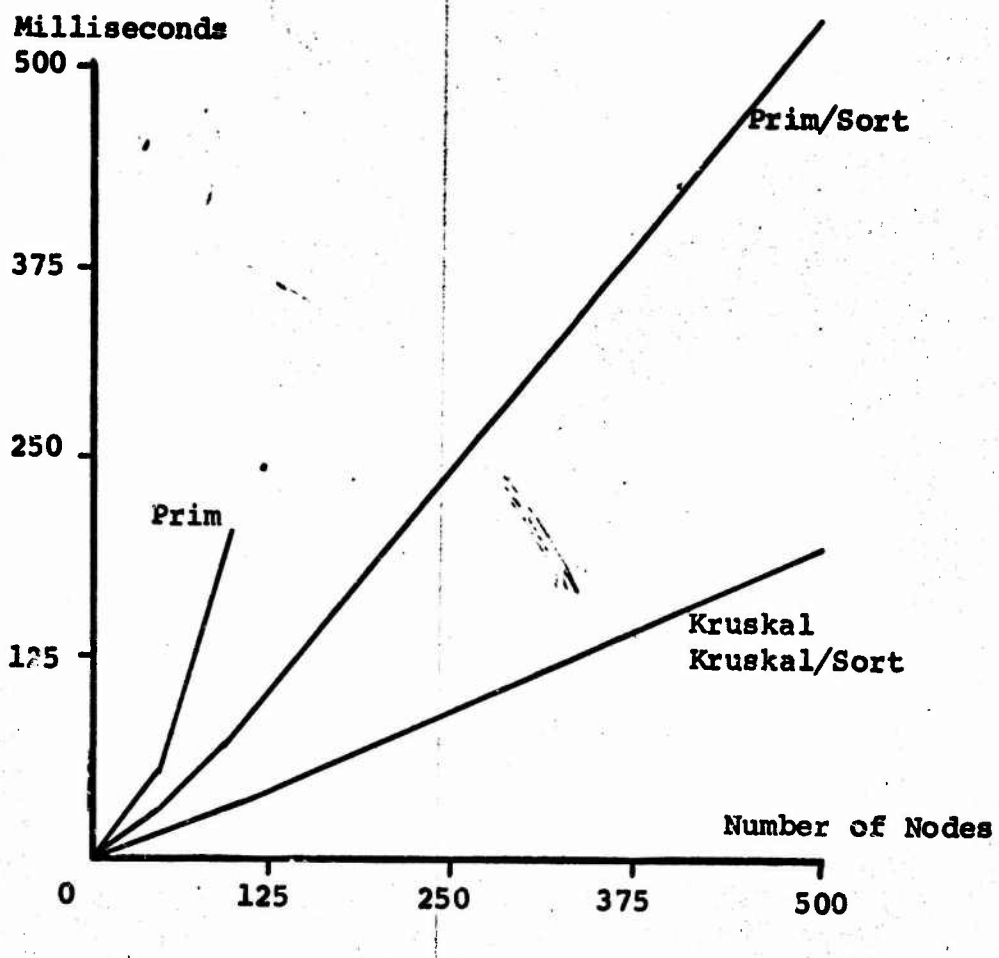


FIGURE 4.1(b)

TIME IN MILLISECONDS TO FIND MST
m=n-1

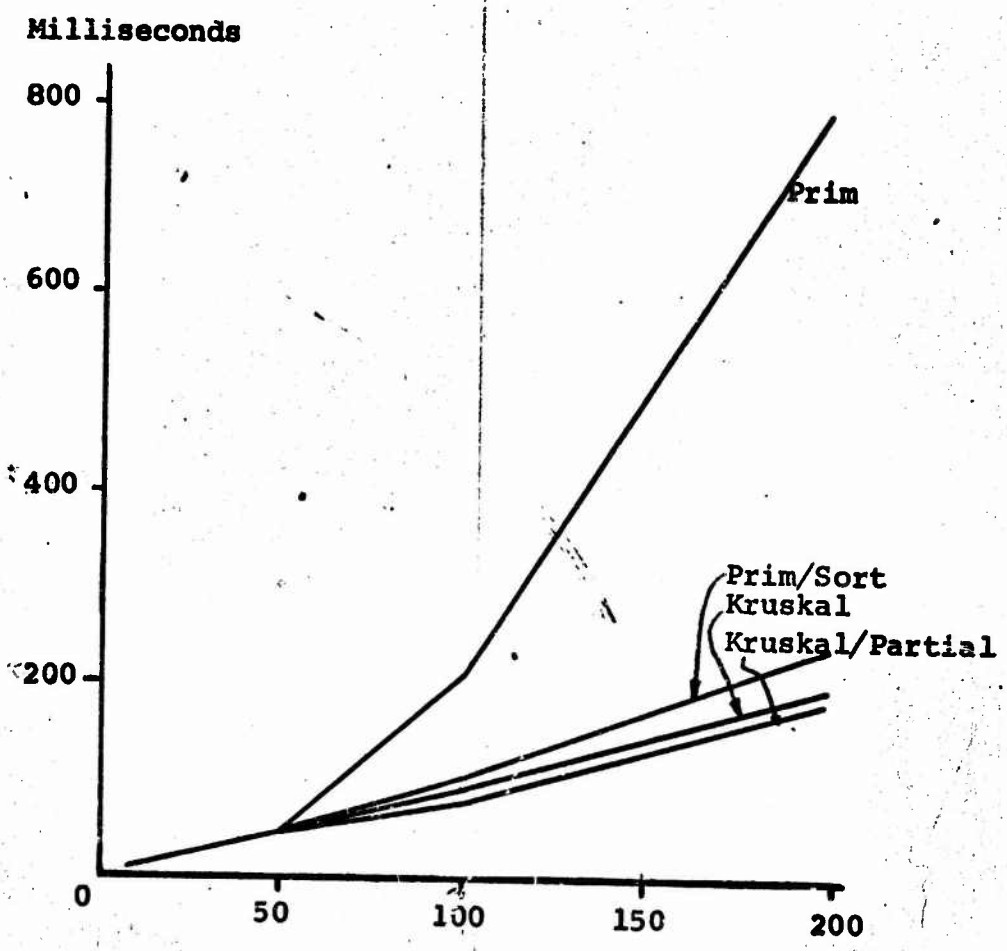


FIGURE 4.1(c)

TIME IN MILLISECONDS TO FIND MST
m=3n

4.3 Shortest Paths in Sparse Networks

Basic principles of the new algorithms are simply stated in the following steps:

1. Degree-one-nodes are removed from the network. For the remaining nodes, nothing has been changed by the removal of these nodes.
2. Degree-two-nodes are removed from the network. Each time a degree-two-node is removed, the distance between its neighboring nodes is replaced by the sum of the distances of the two branches connected to the removed node if the sum is less than the original distance between the two neighbors. This operation will not affect the distances among the remaining nodes.
3. Remaining nodes are examined one by one. A node will be removed if none of its neighbors has been removed in this step. Each time a node is removed, the distance between any two of its neighbors are replaced by the sum of the distances of the two branches connecting the two neighbors to the removed node if the sum is smaller than the original distance. This operation will not affect the distances among the remaining nodes.

4. The following algorithm is then applied to the remaining nodes. Starting from two nodes, shortest paths between all node pairs are determined each time a new node is inserted. The distances between the new node and the rest of the nodes are determined first. For each node, and for each of the new node's neighbors, the sum of the distance from the node to the neighbor and from the neighbor to the new node is obtained. The shortest distances from the new node can then be determined by comparison. The distance between each node pair of the original subnetwork is then calculated. For any node pair, their distance may be shortened only if the shortest paths from the two nodes to the new node contain two different neighbors of the new node, and only if the distance between the two neighbors have been shortened by the introduction of the new node. For a node pair falling into the above category, their distance is replaced by the sum of distances from the two nodes to the new node if the sum is smaller than the original distance.

5. The nodes removed in Steps 3, 2, and 1, are placed back according to the reverse of the sequence in which they were removed. Each time a node is inserted, the distance

between this new node and all the old nodes is adjusted in the same manner as in Step 4.

A flow chart for the algorithm is given in Figure 4.2.

Limitations

It is assumed that there is no cycle of links with negative total weight and that the distance from any node i to any node j is equal to the distance from j to i.

Note: $d(i,j)$ = Distance of the shortest path connecting i and j

$p(i,j)$ = the first intermediate node on an i-j path

a) For any j,k adjacent to N_i,

$$d(j,k) = \text{Min}(d(j,k);\ d(j,n_i) + d(n_i,k))$$

b) Let n be adjacent to $v_1, v_2, \ldots, v_m$.

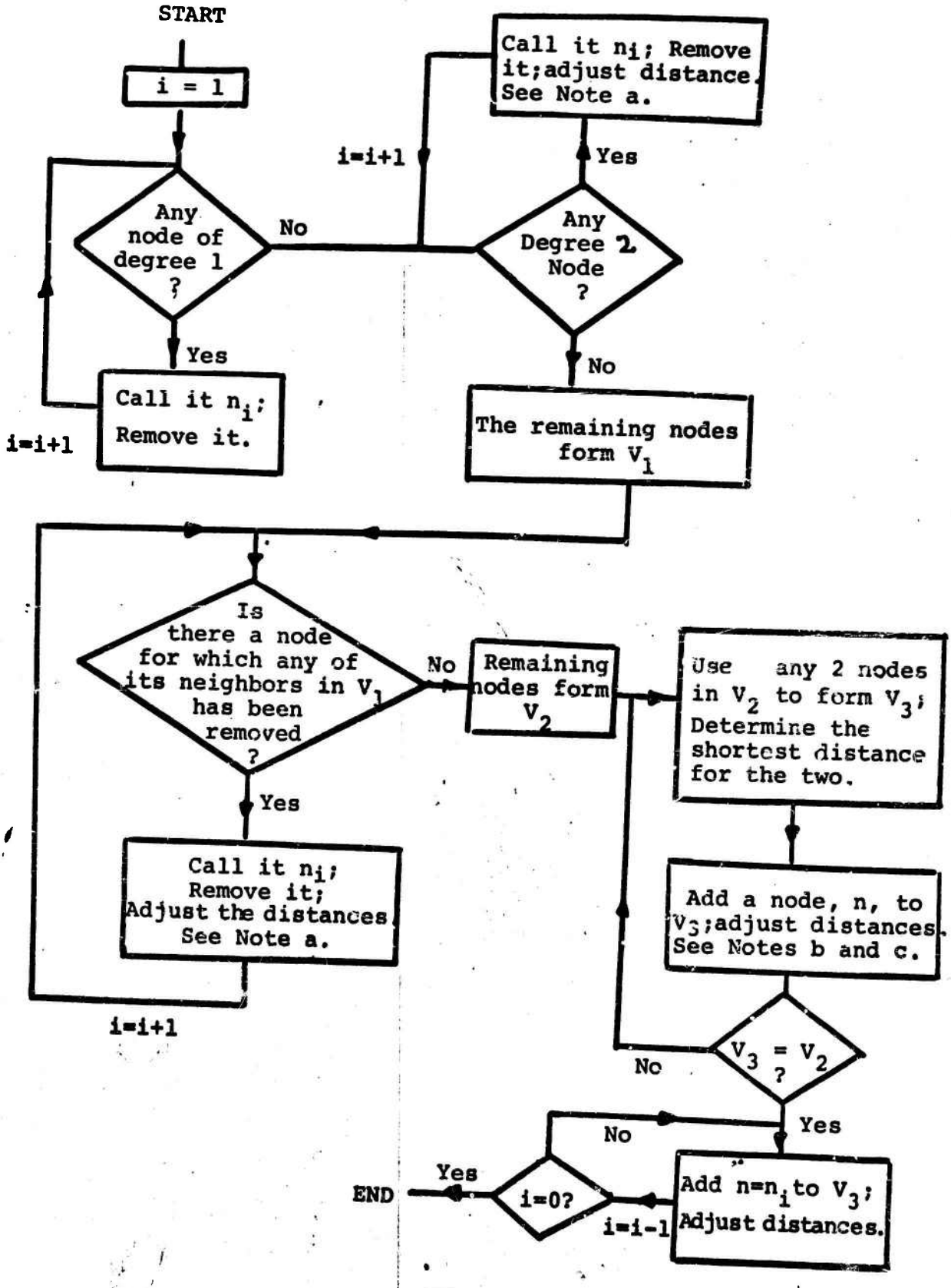
For any $n_3 \in V_3$,

$$d(n,n_3) = \underset{i}{\text{MIN}} \left\{ d(n_3, v_i) + d(v_i, n) \right\}$$

c) For those $i,j \in V_3$ in which $p(n,i) = p(n,j)$ and $d(p(n,k), p(n,j))$ before n is inserted is greater than $d(p(n,i), p(n,j))$ after n is added, $d(i,j) = \text{Min}\ (d(i,j), d(i,n) + d(n,j))$

FIGURE 4.2

FLOW CHART FOR SHORTEST PATH PROGRAM



Numerical Results

Regular networks of 10, 20, 40, 80, 160 nodes which are 2, 3, 4, 5, and 6 connected are generated by the program, and required CDC 6600 computation times to obtain all shortest paths are recorded. Computation times required by using Floyd's Algorithm are also determined. They are listed in Table 4.3.

TABLE 4.3
SHORTEST PATHS COMPUTATION COMPARISON

No. of Nodes	Computer processing time (ms)					
	Floyd's Algorithm	New Algorithm				
		Degree 2	Degree 3	Degree 4	Degree 5	Degree 6
10	10	4	8	10	9	12
20	64	15	25	29	36	43
40	510	56	99	108	154	154
80	4097	205	445	453	784	589
160	32800	788	2086	1747	3692	2310

4.4 Routing Strategies for Computer Network Analysis and Design

A routing strategy for a computer-communication network must define a set of rules to determine the path(s) over which messages should flow from one site to another. There are two different types of routing procedures--those actually implemented in the operating network and those used during the design of the network. A good routing procedure for the design process must be a compromise between three somewhat conflicting requirements: (1) The routing procedure makes full use of available line capacities. This can be interpreted as either minimizing the average delay from message inception to arrival subject to a set of flow requirements or maximizing the throughput subject to a specified maximum delay. (2) The repeated use of the routing procedure during the design stage requires it to be computationally efficient and inexpensive to apply. This usually means that "event type" simulation is impractical. (3) The procedure must be realistic, and similar to the one to be actually implemented in the final operating network.

The objective that the delay be minimized subject to a set of flow constraints makes the routing problem a variation of a nonlinear multicommodity flow problem. This problem,

formulation, discussed in Report 2 and summarized in Section 2, can be readily approached as a separable convex programming problem with the delay as the objective function and the conservation of flow and capacity limitations as the constraints. Minimum delay or maximum throughput can be achieved if the routing procedure follows the solution of the programming problem. However, for networks with more than a few nodes, this approach is extremely expensive for repeated applications of the routing algorithm for use during the design stage.

An analysis of paths selected with the programming approach, yields that most (over 80% in our studies) flow requirements are routed over paths with the minimum number of nodes. A heuristic procedure (described in Report 1 and Report 2) supported by this observation is to route flow over the least utilized such paths. This approach generally gives a result within 5%-20% of optimum. In addition to being fast (over three orders of magnitude faster than the programming approach), routing over paths with only a minimum number of nodes facilitates minor changes of the network structure. On the other hand, this approach can be generalized to produce better results especially for networks with a wide distribution of different line capacities. (An apparently desirable characteristic of very large networks).

The generalization of the basic heuristic approach is described in Report 4. It first routes as much flow as possible over paths with minimum numbers of nodes, (These paths are "shortest paths" using a unit metric for each line.) When no shortest path with excess capacity is available, the saturated lines are deleted from the network and flows are then routed over the shortest paths of the remaining subnetwork. The process is continued until the network is disconnected. By this technique, a message is sent down a path with fewest intermediate nodes and excess capacity, or when that path is filled, the one with next fewest intermediate nodes and excess capacity, etc. The main algorithm is based on "Floyd's Algorithm" with special recognition of the fact that the node degrees in a practical computer network are usually low. The computational savings that are achieved by this recognition are illustrated in Report 4. The new shortest path algorithm discussed elsewhere in this report is also directly useable in this approach. The net effect is a routing procedure which yields flows close to optimal, is physically realistic, and is inexpensive to operate during the design stage.

SUMMARY OF ROUTING PROCEDURES

Routing and Multicommodity Flows

For a computer network, commodities are packets of data to be transmitted through the network. Each node is a terminal as well as a source. Element $r_{k,i}$ in the traffic requirement matrix R can be viewed as the amount of commodity i which is supplied from node k destined for node i. There are then NN commodities where NN is the number of nodes in the network. Each node demands exactly one distinct commodity and supplies NN-1 others. We want to route the traffic such that the average delay time (which corresponds to the cost function) is minimized and each link flow is less than the corresponding link capacity. The average delay time is a non-linear function of the total flow within each line and consequently we must solve a problem considerably more difficult than the classical multicommodity flow problem.

Models for time delay analysis have been described in [KL1], [KL2], and [FR4] and will not be repeated here. If the capacity of the i-th link is C_i and the flow in the link is f_i, and M is the number of links, an effective delay model which accurately predicts delays caused by transmission propagation and queueing, yields the separable convex programming problem:

Model 1.

$$\frac{1}{r}\sum_{i=1}^{M}\left(\frac{(1/\mu)}{C_i - f_i} - \frac{1}{C_i\mu} + \frac{1}{C_i\mu} + P_i + T_{IMP}\right) f_i$$

Subject to

$$Af_i = r_i \qquad \text{for } i = 1, 2, \ldots, NN \qquad (4.1)$$

$$\sum_{i=1}^{N} f_{j,i} - f_j = 0 \qquad \text{for } j = 1, 2, \ldots, M \qquad (4.2)$$

$$\underline{f} \leqslant \underline{C} \qquad (4.3)$$

Constraints (4.1), (4.2), and (4.3) are linear equations with the following interpretations. Constraint (4.1) is called the conservation constraint. This constraint requires that the flow into any node is equal to the flow out of any node. In constraint (4.2), f_j is the total flow in link j. This constraint merely indicates that the flow in any link is equal to the sum of all commodities in that link. The vector $\underline{C}$ in constraint (4.3) is the capacity vector whose j^{th} component is equal to the capacity C_j of the j-th link; $\underline{f}$ is the vector whose j-th component is f_j. Constraint (4.3) requires that the total flow in any link is no greater than the capacity of the link. This

constraint is known as the capacity constraint. Each of the terms in the expression to be minimized can be approximated by straight line segments. The routing problem can then be solved as a linear program. (The number of variables in the linear program is in part a function of the accuracy with which the objective function is approximated. If P linear segments are used in the approximation, then the program has NN(NN-1)+3M constraints and (NN+P+1) M variables.) Also, a parametric approach is able to generate a curve of delay versus traffic with considerable computational savings over solution on a point by point basis.

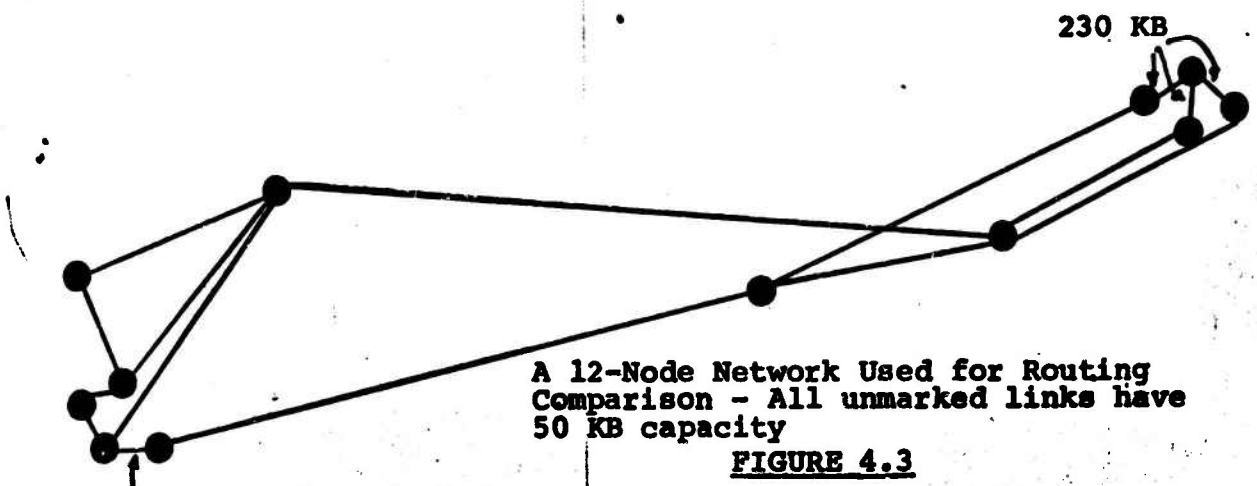
A Basic Minimum Node Routing Algorithm

The total traffic within the network depends on the total input to the network, overhead such as headers, acknowledgments, parity checks, etc., and the numbers of links in the paths chosen for routing. This last factor can be extremely critical, and it can be argued that whenever possible, we should route flow over paths with as few intermediate nodes as possible. A heuristic routing procedure based on this observation requires each message to use a path which contains the lowest number of intermediate nodes from origin to destination.

In the procedure, link capacities can be preassigned or calculated by the algorithm. Traffic is first routed from node i to any node j, which is directly connected to i, over link (i,j). Consequently, after this stage, some flows have been assigned to the network. Each node connected to i by minimum node paths with one intermediate node is then considered. For any node j in this group, all feasible paths from i to j are examined, and functions of the flow thus far assigned to each link and the required capacity (or preassigned capacity) for this flow are evaluated. We then select paths which have minimum "resistance" to additional flow (e.g., paths whose maximum path flow is minimum, paths with the largest minimum residual capacity, or paths with the smallest maximum link utilization.). From the subset of paths thus selected, the path whose total physical length is minimum is then chosen and all traffic originating at i and destined for j is routed over this path.

Examples of the above routing procedure applied to a 12-node network shown in Fig. 4.3 are given in Figures 4.4 and 4.5 Figure4.4has equal traffic requirements between all node pairs while Figure 4.5 has random traffic requirements. Thecurves marked "optimal" represent the results of the multicommodity flow approach discussed in the last section.

The routing procedure as discussed above deals with the network whose structure has been specified. Optimization of the network's structure often involves the repeated analysis, modification, and re-analysis of proposed structures. Network modifications consist of additions or deletions of links either one or more at a time. Each time a modification is made, the routing and analysis algorithms must be applied to determine the cost and feasibility of the new network. Special routing algorithms can be deduced from the basic algorithm to reduce the computation time at each step. The basic algorithm operates by assigning numbers called "labels", to each node. From these labels, routes are calculated for the overall network. The special algorithms operate by examining these labels before a change in the network structure is made and calculating the effects of the change without recomputing all of the labels. These algorithms make critical use of the properties of minimum node (or shortest) paths via basic "triangle" inequalities. As an example, suppose a link is added from node x to node y. Let L(u,v) be the number of nodes in a minimum node path between any nodes u and v. Then, obvously the minimum node paths between node i and j remain invariant if, and only if, $L(i,x) + L(y,j) \geq (L(i,j)-1)$. If $L(i,x) + L(y,j) = L(i,j)-1$, then the original



A 12-Node Network Used for Routing Comparison - All unmarked links have 50 KB capacity

FIGURE 4.3

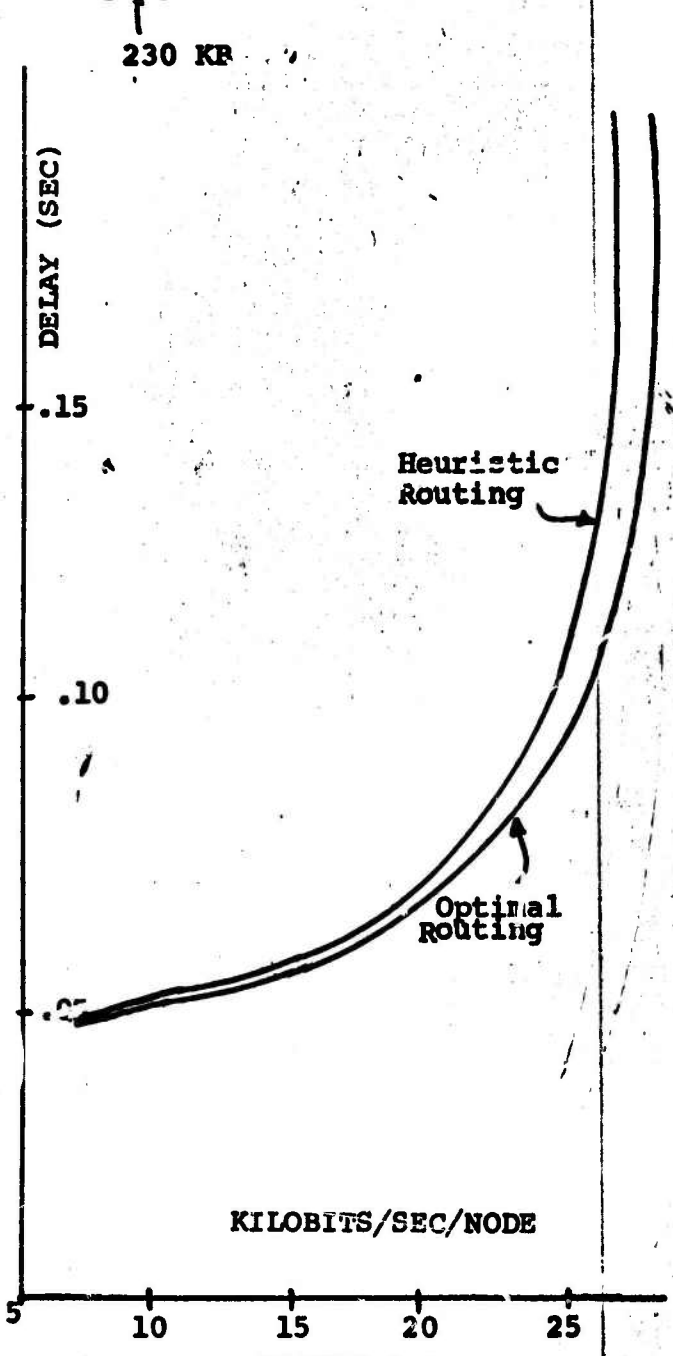


FIGURE 4.4
12-NODE UNIFORM TRAFFIC

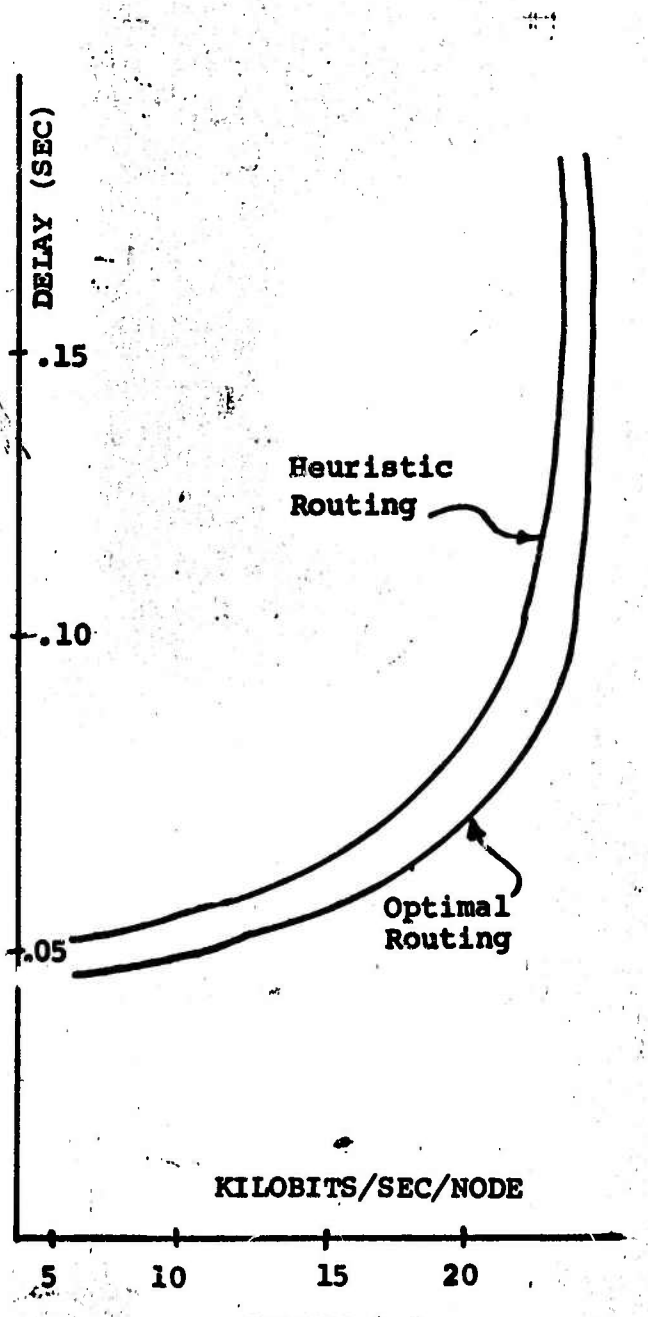


FIGURE 4.5
12-NODE RANDOM TRAFFIC

minimum path has increased by at least one. If $L(i,x) + L(y,j) < L(i,j)-1$, all new minimum node paths between i and j contain the link (x,y). These paths may be generated by concatenating the set of minimum node paths between node i and x with the link (x,y) and then with the set of minimum node paths from y to j. In any event, the new node labels are obtained by inspection and all new minimum node paths can be generated from the known minimum paths which existed before link (x,y) was added.

A Cut-Saturation Routing Technique

The heuristic minimum node routing strategy yields a near optimal solution (usually within 5% to 20%). This algorithm is especially effective during the network optimization process as stated in the last section. However, it has two drawbacks when it is used for the sole purpose of traffic routing. First, all minimum node paths between each node pair are generated while only one path is used. Computer time may be saved if only the paths to be actually used are generated. Second, the routing process terminates if any one link is saturated. Higher throughputs could be obtained if alternate paths are then used and the algorithm terminated when a cut (i.e. a set of links whose removal disconnects the network) is saturated.

The routing algorithm described below provides the two improvements required. Based on the new algorithm described above, a single minimum node path (hereafter called a shortest path) is generated between each pair of nodes. Note that with the new algorithm all shortest paths are generated simultaneously. The required traffic is then routed over the unique path for each node pair. The traffic flow between each node pair and on each link is either uniformly (or in special cases, non-uniformly) increased or decreased until the flow is equal to the capacity for the most utilized link. Here we have assumed that link capacities are preassigned. The saturated link(s) is then removed from the network and the capacity on each link is replaced by its residual capacity at this point. A shortest path is again generated for each node pair and additional traffic is routed as before. The process is repeated until the network is disconnected or until all required traffic has been sent. Although the approach is heuristic, the result is so close to optimal for high traffic that no significant difference can be observed for the networks studied.

The first iteration of the cut-saturation routing algorithm (one application of the shortest path algorithm) enables us to route along shortest paths about as much flow as the minimum

node method. Thus, one may initially expect throughputs within 5-20% of optimal. Saturated links may then be removed one or more at a time and the complete process repeated. After each such iteration, more traffic can be sent through the network.

As an example, the 23 node network shown in Figure 4.6 has five saturated links. Five iterations were required to obtain these flows before enough links were removed to disconnect the network. The performance of the network after each iteration is plotted as throughput versus delay curves in Figure 4.7.

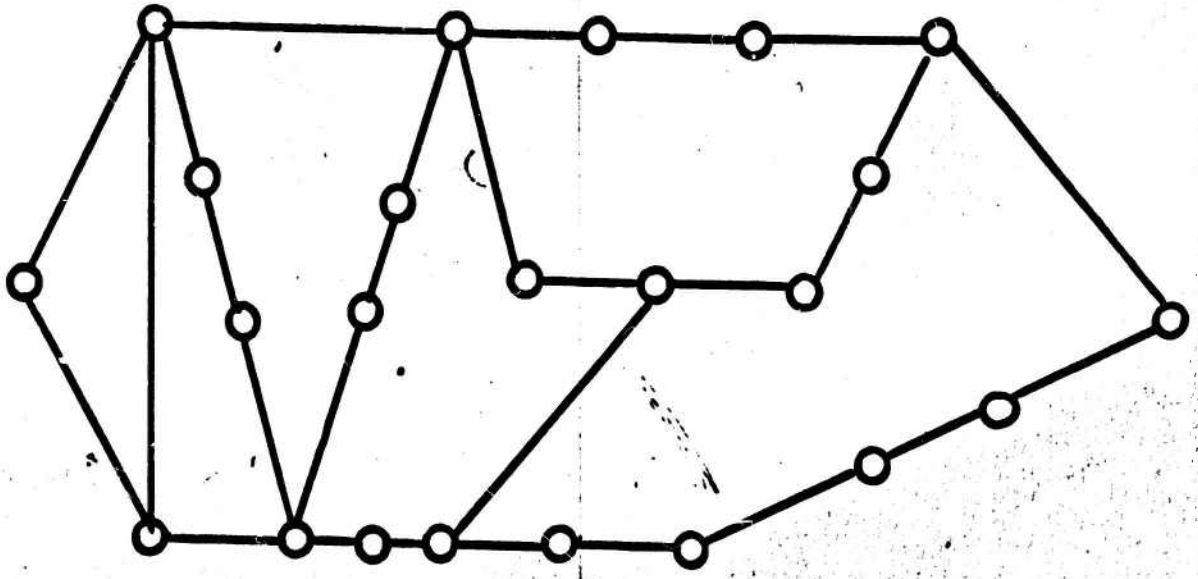


FIGURE 4.6

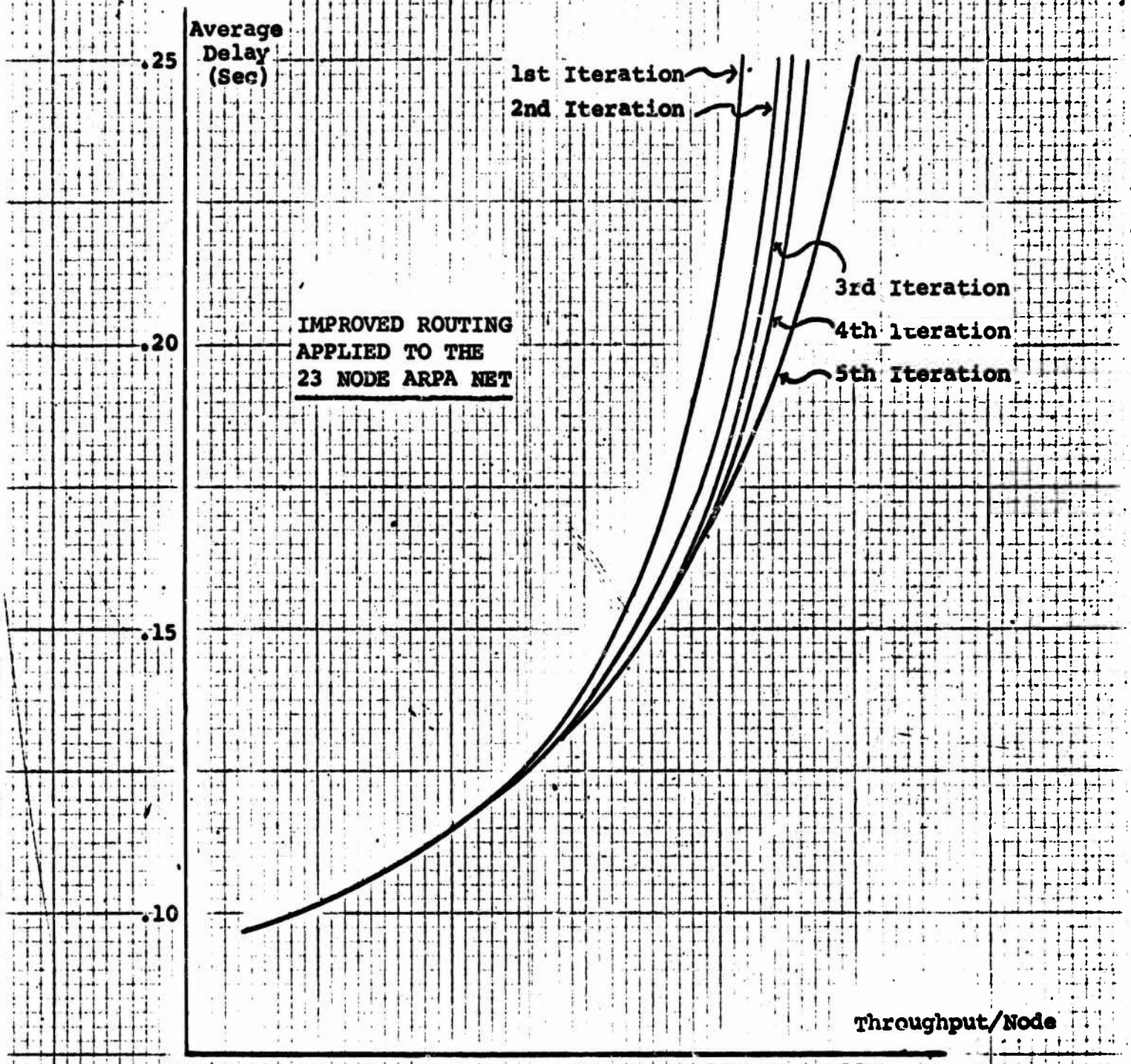

Average Delay (Sec)
.25
.20
.15
.10
1st Iteration
2nd Iteration
3rd Iteration
4th Iteration
5th Iteration
IMPROVED ROUTING APPLIED TO THE 23 NODE ARPA NET
Throughput/Node
6K
7K
8K
9K
10K
11K

FIGURE 4.7

4.5 Network Reliability Analysis

General Structures

The analysis approach described is based on the use of an analytic expression for reliability analysis and was first discussed in Report 3. While in theory, this analytic expression completely describes the behavior of the system, all of the terms in the expression cannot be calculated analytically in a practical sense, and thus, the expression is only useful as a model.

The model is most easily described when all nodes and all links have common probabilities of failure. A special case, all nodes are completely reliable, is especially easy to discuss. For this case, we can write

$$F(p) = \sum_{k=0}^{B} C_F(k)\ p^{B-k}\ q^k$$

where F(p) is the probability the network will fail, $C_F(k)$ is the number of subnetworks which have failed (contained in the original network) with k surviving links, p is the common link failure probability, q=1-p, and B is the total number of links in the network. For example, if we wish to compute the probability that all nodes can communicate, then $C_F(k)$ is the number of disconnected subnetworks containing k links. If we wish to

compute the probability that a specified pair of points A and B can communicate, then $C_F(k)$ is the number of subnetworks with k links that do not have at least one surviving path between A and B.

The difficulty of computing all of the $C_F(k)$ depends on the definition of network failure. However, for all but the most trivial definitions, this computation is not possible analytically for all k. On the other hand, a good deal of partial information is available. Thus, if C(k) is the number of disconnected k link networks, and N is the total number of nodes in the network,

$$C(k) = \binom{B}{k} \text{ for } k = 0, 1, 2, \ldots, N-1$$

and

$$C(k) = 0 \quad \text{for } k = B-\tau + 1, \ldots, B$$

Here, τ is the size of the minimum link cutset.* Also, if $C_{a,b}(k)$ is the number of subnetworks containaing k links in which points a and b cannot communicate, then

$$C_{a,b}(k) = \binom{B}{k} \text{ for } k = 0, 1, \ldots, L(\pi_{a,b}) - 1$$

and

$$C_{a,b}(k) = 0 \quad \text{for } k = B - \tau_{a,b}+1, \ldots, B$$

---

* A link cutset is a set of links whose removal from the network breaks all paths between at least one pair of nodes.

In this expression, $L(\Gamma_{a,b})$ is the number of links in a path with the fewest number of links between points a and b, and $\gamma_{a,b}$ is the size of the minimum link cutset separating a and b. As a concrete example, consider the network shown in Figure 4.8. This network has 8 nodes and 12 links. The minimum cutset in the network has 2 links and the minimum cut separating a and b has 3 links. Moreover, every path between A and B has at least 3 links. Therefore,

$$C(k) = \binom{12}{k} \quad \text{for } k = 0, 1, 2, \ldots, 7$$

$$C(k) = 0 \quad \text{for } k = 11, 12$$

$$C_{a,b}(k) = \binom{12}{k} \quad \text{for } k = 0, 1, 2$$

$$C_{a,b}(k) = 0 \quad \text{for } k = 10, 11, 12$$

The remaining $C_F(k)$ are not as readily computed. These terms can be calculated or estimated by enumeration, in some special cases by formula, or in general by simulation.

The calculation by simulation of the unknown $C_p(k)$ can be combined with a simulation procedure known as stratification. To discuss stratification, we return to the case where both nodes and links may fail but each has a common failure probability. Let p_1 be the probability that any given node is killed and let

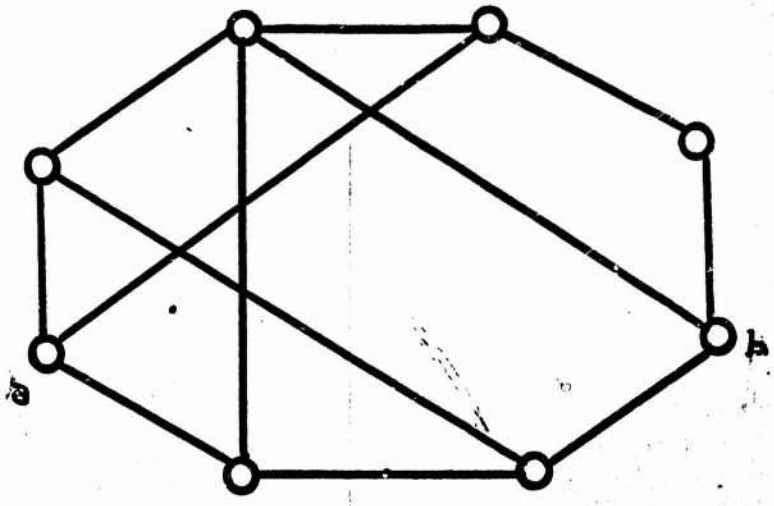


FIGURE 4.8

p_2 be the probability that any given link is killed. The strata are defined by the number of links destroyed and the number of nodes destroyed. Thus, the first stratum is defined by 0 links destroyed - 0 nodes destroyed, the second by 1 link destroyed - 0 nodes destroyed, and the next several strata by 0 links destroyed - 1 node destroyed, 2 links destroyed - 0 nodes destroyed, 1 link destroyed - 1 node destroyed, 0 links destroyed - 2 nodes destroyed.

A network with exactly m destroyed links and n destroyed nodes has probability of occuring

$$\binom{B}{m}\binom{N}{n}\ (1-p_1)^{N-n}\ p_1^{\,n}\ (1-p_2)^{B-m}\ p_2^{\,m}$$

Proportional random sampling is then used to divide the effort devoted in the simulation according to the probability of each stratum.

As an example, again consider the network of Figure 4.8. Suppose the probability of any element failing (either node or link) is 0.05. The probabilities of the first few strata are then:

$$\binom{12}{0}\binom{8}{0}(.05)^0(.95)^8(.05)^0(.95)^{12} = .358$$

$$\binom{12}{0}\binom{8}{1}(.05)^1(.95)^7(.05)^0(.95)^{12} = .226$$

$$\binom{12}{1}\binom{8}{2}(.05)^2(.95)^6(.05)^1(.95)^{11} = .151$$

$$\binom{12}{2}\binom{8}{0}(.05)^0(.95)^8(.05)^2(.95)^{10} = .0655$$

$$\binom{12}{1}\binom{8}{0}(.05)^1(.95)^7(.05)^1(.95)^{11} = .0950$$

$$\binom{12}{0}\binom{8}{2}(.05)^2(.95)^6(.05)^0(.95)^{12} = .0282$$

When we approach the stage of simulation, we first note that we can express our failure probability as

$$F(p_1,p_2) = \sum_{k_1=0}^{N} \sum_{k_2=0}^{M} C_F(k_1,k_2)\, p_1^{N-k}\, q_1^{k}\, p_2^{B-k}\, q_2^{k}$$

where $C_F(k_1, k_2)$ is the number of <u>failed</u> subnetworks containing k_1 nodes and k_2 links. That is, $C_F(k_1, k_2)$ is the number of failed subnetworks in the stratum defined by $N-k_1$ nodes destroyed, $B-k_2$ links destroyed.

We next note that a number of the $C_F(k_1, k_2)$ are known <u>a priori</u>. Thus, to compute the probability of communication between all points,

$$C_F(k_1,k_2) = \binom{B}{k_2} \text{ for } k_2 = 0, 1, \ldots, k_1 - 1$$

$$C_F(k_1,k_2) = 0 \qquad \text{for } k_1 = B-\omega+1, \ldots, B$$
$$\text{or } k_2 = B-\tau+1, \ldots, B$$

We therefore need not devote any computation to these coefficients.

As our next step, we generate random variables, delete links and nodes, examine the connectivity of the resulting system for the desired properties, and record each outcome. Using the stratification principle, and proportional random sampling, we sample from each unknown stratum in proportion to its probability of occurance. In our example, to compute the message delivery probability between points a and b, we need not sample any stratum which has a total of less than three failed nodes and/or links, because the minimum cut separating a and b has 3 elements and at least 3 elements must fail in order to break communications between a and b.

If link and node failure probabilities are either very high or very low, the stratification approach leads to computational savings of several orders of magnitude. In fact, this procedure is far superior to any other one for very large networks which have either very low or very high element failure probabilities. The proportional sampling approach is well suited

even to the case where each link and node failure probability is a possibly different number, as long as there is not a wide variation among failure probabilities.

If wide variations among element failure probabilities occur, stratified sampling is still effective but the use of proportional sampling is no longer appropriate. This problem is caused by the fact that the probability of a stratum can no longer be calculated by the number of ways an event in the stratum could occur multiplied by the probability of any such event. Instead, explicit enumeration of combinations of link and node probabilities is required at a substantial increase in computation time. This approach is therefore no longer practical. However, the stratification procedure is still useful and the savings in complexity created by our a priori knowledge of some of the $C_F(k_1, k_2)$ are still achieved. The only additional requirement is the allocation of a sample size to each unknown stratum. Many approaches are possible. One could, for example, allocate an equal number of samples to each stratum. Once a particular stratum is chosen for a step in the simulation, the specific links and nodes to be deleted could then be selected by proportional sampling and the analysis then continue as described above.

A comparison between the stratification and the usual simulation approach for a 23 node ARPANET (Fig. 4.9) is given in Table 4.4.

Special Structures

The network structure of many common communication networks can be represented as a composite of simple loops and trees. Reliability analysis of such networks can be carried out very quickly and efficiently by a new recursion approach described in Report 5. Moreover, a wide variety of reliability measures can be obtained using the same general method. The measures studied in Report 5 are:

(i) the expected number of nodes communicating with a central node called a "root",

(ii) the expected number of node pairs communicating,

(iii) the expected number of node pairs communicating by a path through the central node,

(iv) the probability that operating nodes can communicate through the root,

(v) the probability that operating nodes are connected.

Many other measures are possible.

In Fig. 4.10 some of the many network structures that can be analyzed [illegible] recursion are illustrated. In addition, even if a network does not have this precise structure, the reliability of the network can often be approximated by the reliability of

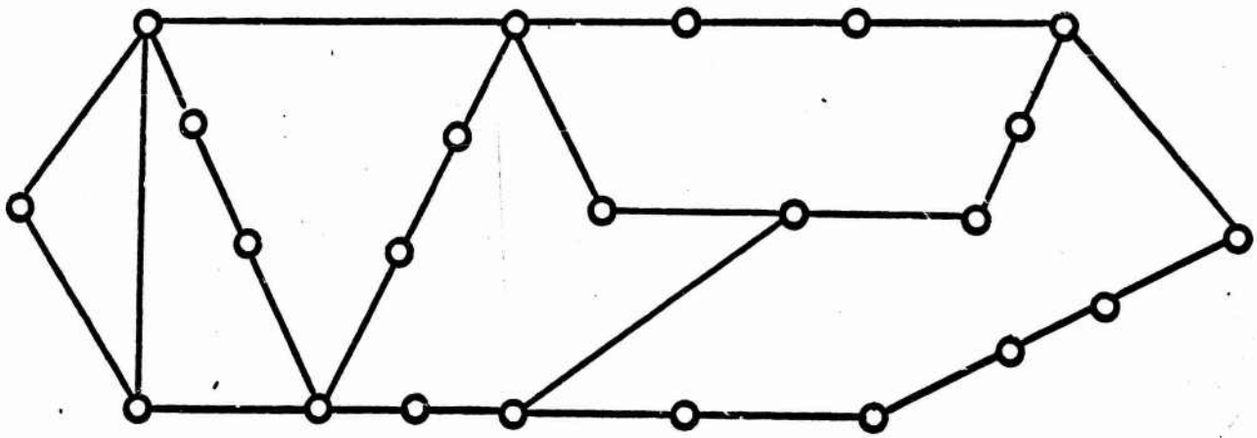


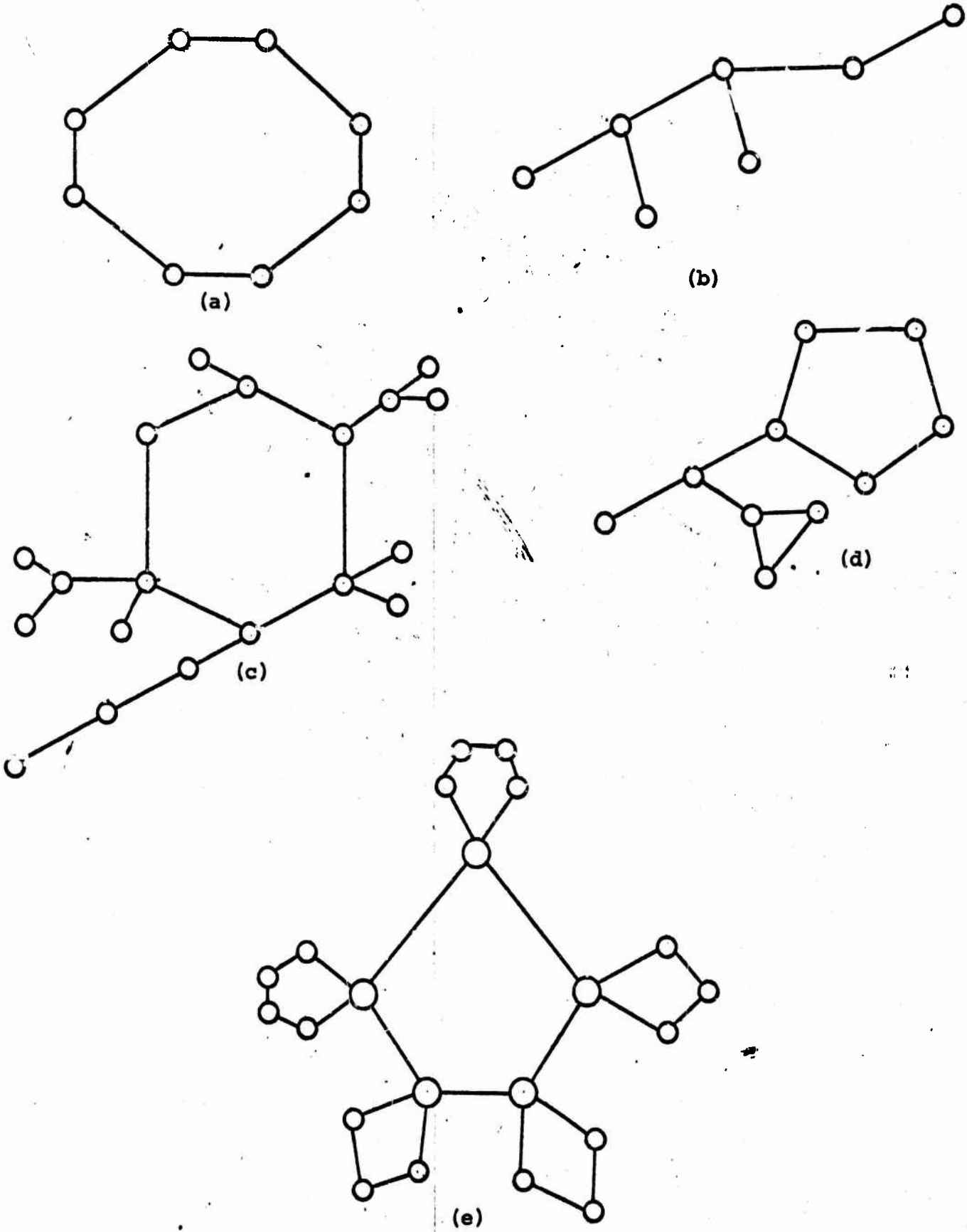
FIGURE 4.9

TABLE 4.4
COMPARISON OF TWO DIFFERENT SIMULATION METHODS
Sample Size 1000

Link Failure Probability	Probability of Disconnected Net		Standard Deviation	
	Stratified Sampling	Straightforward Sampling	Stratified	Straightforward
.01	.00297	.004	5.15×10^{-5}	1.99×10^{-3}
.02	.01178	.012	3.28×10^{-4}	3.44×10^{-3}
.03	.02625	.027	8.96×10^{-4}	5.12×10^{-3}
.04	.04613	.042	1.74×10^{-3}	6.34×10^{-3}
.05	.07109	.070	2.85×10^{-3}	8.06×10^{-3}
.06	.10075	.097	4.17×10^{-3}	9.35×10^{-3}
.07	.13466	.135	5.66×10^{-3}	1.08×10^{-2}
.08	.17230	.178	7.29×10^{-3}	1.20×10^{-2}
.09	.21312	.224	8.99×10^{-3}	1.31×10^{-2}
.1	.25654	.276	1.07×10^{-2}	1.41×10^{-2}
.2	.70384	.743	1.60×10^{-2}	1.38×10^{-2}
.3	.93931	.954	5.30×10^{-3}	6.62×10^{-3}
.4	.99382	.995	6.32×10^{-4}	2.23×10^{-3}
.5	.99971	.999	2.89×10^{-5}	$1. \times 10^{-3}$
.6	.99999	1.000	4.23×10^{-7}	0

FIGURE 4.10

COMPOSITE LOOP AND TREE STRUCTURES



such a network or a hybrid computation using recursion on the tree and loop parts of the network together with simulation for the other parts can be carried out. These techniques thus offer a very powerful tool in the analysis of network reliability.

Terminology.

We will develop a very general class of recursive methods for a wide variety of reliability criteria. To do this it is very economical to employ a recursive characterization of rooted trees.

Definition: A rooted tree is a finite set T of one or more nodes such that:

a) There is one specially designated node called the root of the tree, root (T); and

b) The remaining nodes (excluding the root) are partitioned into $m \geq 0$ disjoint sets T_1, T_2, T_3, ..., T_m, and each of these sets in turn is a rooted tree. The trees T_1, ..., T_m are called subtrees of the root.

The root, J, of a tree is said to be the father of the root of each of the subtrees of J. The root, I, of a subtree of J is said to be a son of J. Fig.4,11 depicts such a rooted tree graph where links are shown between fathers and their sons.

A link is a pair of nodes one of which is the father of the other. Thus node 1 is the root of the entire tree. Node 2 is the root of the only subtree of 1 and hence 2 is the son of 1 and 1 the father of 2. The corresponding subtree of 1 is determined by the nodes $\{2, 3, 4, 5, 6, 7, 8, 9, 10\}$. Node 2 has two subtrees on $\{3, 4, 5\}$ and $\{6, 7, 8, 9, 10\}$ with roots 3 and 6 respectively. Node 3 has two subtrees $\{4\}$ and $\{5\}$. Node 4 has no subtrees.

Since we will be dealing with computer methods of solution, it is necessary to impose a linear ordering for storage purposes. This will be done by a father function. Suppose we have a network on NN nodes, $\{1, 2, \ldots, NN\}$, and for each node I except 1 we have a node F(I), the father of I, such that $F(I) < I$ and (I,F(I)) is a link in the network. Then F defines NA = NN - 1 links and in fact the existence of a father function F is a necessary and sufficient condition for the network to be a rooted tree. The special node 1 (which has no father) is of course the root of the tree (sometimes called the patriarch). Associated with each node I is a rooted subtree consisting of nodes with greater numbers which are connected to I by a path passing through nodes with labels $\geq I$. In Table 4.5 the father-function for the tree in Fig. 4.11 is given.

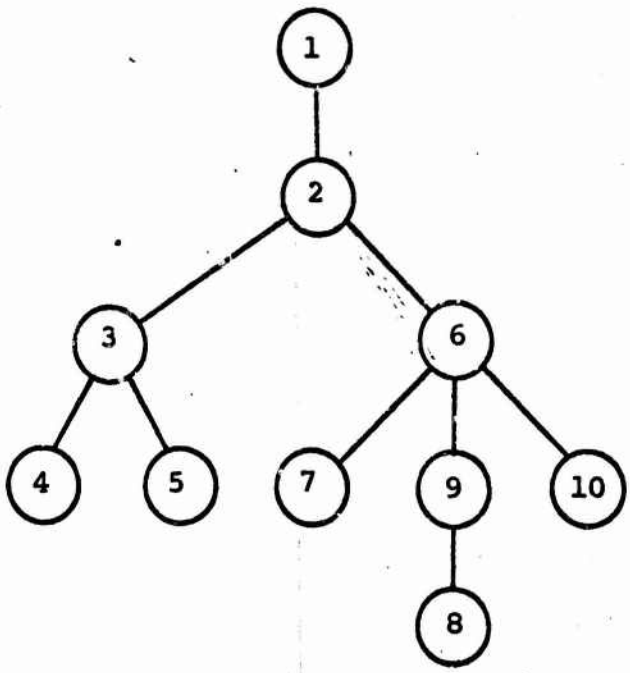


FIGURE 4.11

I	F(I)
1	-
2	1
3	2
4	3
5	3
6	2
7	6
8	7
9	6
10	6

TABLE 4.5
FATHER FUNCTION

Recursive Computations on Trees

We now want to calculate the reliability of a tree network assuming the reliability of its elements, nodes and links, are known. It is not immediately obvious what the "reliability of a tree" should mean; we will consider several meanings. However, the general approach in each case will be the same. Considering the tree to be a rooted tree, we associate a state vector with the root of each of the subtrees. The state vector associated with a root node contains all information about that node relevant to our calculation, for example, the expected

number of nodes which can communicate with that node and the expected number of node pairs communicating in the subtree rooted at that node. Other examples of state vectors are given in Table 4.6. We then define a set of recursion relations which yield the state vector of a rooted tree given the state of its subtrees. For subtrees consisting of single nodes the state is obvious. We then join the rooted subtrees into larger and larger rooted subtrees using the recursion relations until the state of the entire network is obtained.

Deriving the recurrance relations is somewhat mechanical also. It comes simply from considering the situation depicted in Figure 4.12. We have two subtrees, one with root I and the other having as its root J = F(I). We assume the state of I and J are known and we wish to compute the state of J relative to the tree obtained by joining I and J by the link (I,J).

To illustrate the technique let us consider the first and easiest criterion. Namely we wish to know the expected number of nodes which can communinate with the root node 1. We assume we have associated with each node I a probability of node failure PN(I) and a probability QN(I) = 1 - PN(I) of the node being present. Similarly for the link (I,F(I)) we have probabilities PL(I) and QL(I) of the link failing and being operative,

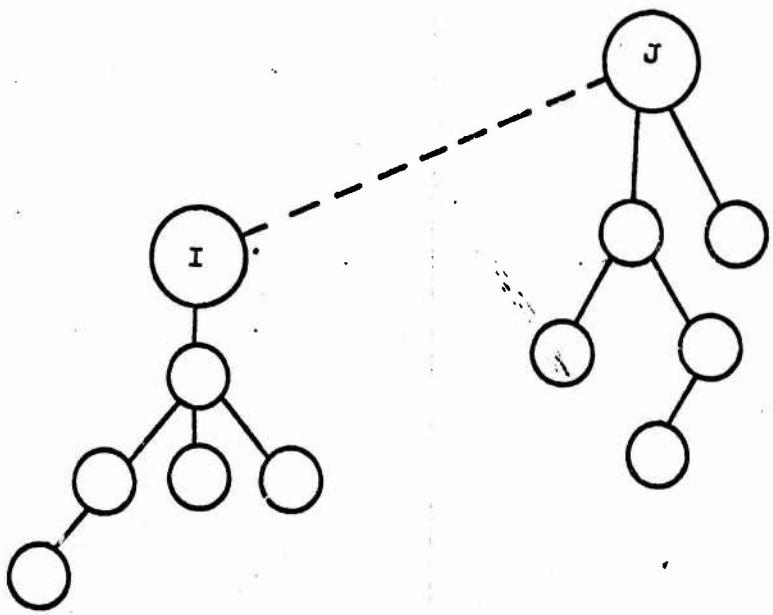


FIGURE 4.12

respectively. The state vector of a subtree with root I, is in this case a scalar, S(I), which is the expected number of nodes in the subtree which communicate with the root I, including I. To derive the recurrence relations we consider two subtrees with I and J = F(I) as roots, respectively. We then want to derive the state of the new subtree obtained by joining I and J together by (I,J). Let S(I) and S(J) be the known states for the two subtrees and S(J)' the resulting state. If the link (I,J) and the node J are operational S(J)' = S(I) + S(J); if not, then S(J)' = S(J). Putting the two together we have the recurrence relation: S(J)' = S(J) + S(I) QN(J) QL(I) where QN(J) is the probability that node J is operative and QL(I) is the probability that the link (I,J) is operative. Now all that remains is to put this in the form of an algorithm:

Step 0 (Initialization). Set S(I) = QN(I) (the probability that node I is working), I = 1, ..., NN. Set I = NN. Go to Step 1.

Step 1: Let J = F(I), and set S(J) to S(J) + S(I) QN(J) QL(I), go to Step 2.

Step 2: Set I to I - 1. If I = 1, stop; otherwise, go to Step 1.

When the algorithm stops S(1) is the expected number of nodes communicating with 1 counting 1.

We now examine a class of reliability criteria related to whether the network is connected or not. The first result is immediate: The probability QC of the tree being connected is

$$QC = \prod_{1}^{NN} QN(I) \prod_{2}^{NN} QL(I) \qquad (4.4)$$

If we do not insist that the entire network be connected but only the subnetwork involving operative nodes be connected we get a new probability $\tilde{QC}$. The calculation is more interesting in this case. Here we need a state vector for each subtree with 3 components. They are:

N(I) - The probability that all nodes in the subtree are failed.

C(I) - The probability that the (non-null) set of operative nodes, including the root of the subtree, are connected.

B(I) - The probability that the root of the subtree is failed and the set (non-null) of operative nodes in the subtree is connected.

N(I), C(I), and B(I) account for all tree networks whose operative nodes communicate.

The recurrence relations in this case are

$$C(J)' = C(I)\ C(J)\ QL(I) + C(J)\ N(I) \qquad (4.5a)$$

$$N(J)' = N(I)\ N(J) \qquad (4.5b)$$

$$B(J)' = B(J)\ N(I) + B(I)\ N(J) + C(I)\ N(J) \qquad (4.5c)$$

As we mentioned before, often in practical situations all communication has to take place through the root node. So another interesting reliability condition is the probability, QR, that all operating nodes can communicate with the root. As can be seen from the definition of C, QR = C(1) + N(1).

An algorithm for obtaining both criteria is:

Step 0: (Initialization) Set N(I) = PN(I), C(I) = QN(I), B(I) = 0, I = 1, ..., NN. Set I = NN. Go to Step 1.

Step 1: Let J = F(I). Using Equations (4.5a), (4.5c), (4.5c), recalculate B(J), C(J), and N(J), in that order. (Note that the order of calculations is important as calculations should be done with the old values of B(J), C(J), and N(J)). Go to Step 2.

Step 2: Set I = I - 1. If I = 1, stop; otherwise go to Step 1.

After the algorithm terminates, we obtain the probability of all operating nodes communicating by $\widetilde{QC}$ = C(1) + B(1) + N(1) and the probability of all operating nodes communicating with the root by QR = C(1) + N(1).

We summarize the various algorithms in Table 4.6. The algorithms for finding the reliability measures discussed in this section were coded in FORTRAN IV and executed on a CDC-6600.

The average running time for a 500 node tree was 1.5 seconds. In Report 5, approaches for analyzing trees with weighted nodes and looped networks are also discussed.

TABLE 4.6

SUMMARY OF RELIABILITY MEASURES

Measure	Given By	State Vector
Exp(number of nodes communicating with root)	S(1)	S
Exp(number of node pairs communicating)	T(1)	S,T
Exp(number of node pairs communicating through root)	R(1)	S,R
Prob(operating nodes can communicate through root)	N(1)+C(1)	N,C
Prob(operating nodes are connected)	N(1)+C(1)+B(1)	N,C,B

Component	Recursion Relation	Initial Condition
S	$S(J)'=S(J)+S(I)QN(J)QL(I)$	$S(I)=QN(I)$
T	$T(J)'=T(J)+T(I)+S(I)S(J)QL(I)$	$T(I)=0$
R	$R(J)'=R(J)+(R(I)QN(J)+S(I)S(J))QL(I)$	$R(I)=0$
N	$N(J)'=N(I)N(J)$	$N(I)=PN(I)$
C	$C(J)'-(C(I)QL(I)+N(I))C(J)$	$C(I)=QN(I)$
B	$B(J)'=B(J)N(I)+B(I)+C(I))N(J)$	$B(I)=0$

5. PUBLICATIONS RESULTING FROM RESEARCH EFFORT

1. H. Frank, I. T. Frisch, W. Chou, "Topological Considerations in the Design of the ARPA Computer Network," Proceedings of the SJCC, 1970

2. H. Frank, I. T. Frisch, R. Van Slyke, W. Chou, "Optimal Design of Centralized Computer Networks," Proceedings of the International Conference on Communications, June, 1970, and Networks, Volume 1, No. 1, 1971

3. H. Frank, W. Chou, "Routing in Computer Networks," Networks, John Wiley, 1971, Vol. 1, No. 2, pp. 99-112

4. H. Frank, W. Chou, "Throughput in Computer-Communication Networks," International Report on the State of the Art of Computer Networks, Infotech, London, 1972

5. R. Van Slyke, H. Frank, "Reliability of Computer-Communication Networks," Proc. of the Fifth ACM Conference on Applications of Simulation, New York, December, 1971

6. R. Van Slyke, H. Frank, "Network Reliability Analysis--I," Networks, Vol. 1, No. 3, 1972

7. W. Chou and H. Frank, "Routing Strategies for Computer Network Design," Proceedings of the Brooklyn Polytechnic Symposium on Computer Networks, April, 1972

8. H. Frank, R. Kahn, L. Kleinrock, "Computer Communication Network Design--Experience With Theory and Practice," Proceedings of the SJCC, 1972

9. A. Kershenbaum and R. Van Slyke, "Computing Minimum Spanning Trees Efficiently," Proceedings of the ACM Annual Conference, August, 1972

10. H. Frank, W. Chou, "Topological Optimization for Computer Networks," Proceedings of the IEEE, November, 1972

11. A. Kershenbaum, R. Van Slyke, "Recursive Analysis of Network Reliability," Networks, January 1973

12. H. Frank, R. Van Slyke, "Reliability and Large Computer Networks," in preparation.